

RBSE: Product Development Team Research Activity Deliverables

GHG Corporation

1992

GRANT
1N-61-CR
167276
p- 288

**Cooperative Agreement NCC 9-16
Research Activity No. RB.05**

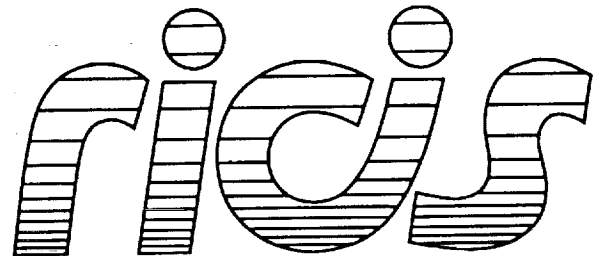
**NASA Technology Utilization Program
NASA Headquarters**

N93-29212

Unclass

G3/61 0167276

(NASA-CR-193115) RBSE: PRODUCT
DEVELOPMENT TEAM RESEARCH ACTIVITY
DELIVERABLES (Research Inst. for
Computing and Information Systems)
288 p



*Research Institute for Computing and Information Systems
University of Houston-Clear Lake*

***RBSE: Product Development Team
Research Activity Deliverables***

RICIS Preface

This research was conducted under auspices of the Research Institute for Computing and Information Systems by GHG Corporation. Jeffrey O. McGee was GHG Product Development Project Manager. Dr. E. T. Dickerson served as RICIS research coordinator.

Funding was provided by the NASA Technology Utilization Program, NASA Headquarters, Code C, through Cooperative Agreement NCC 9-16 between the NASA Johnson Space Center and the University of Houston-Clear Lake. The NASA research coordinator for this activity was Ernest M. Fridge III, Deputy Chief of the Software Technology Branch, Information Technology Division, Information Systems Directorate, NASA/JSC.

The views and conclusions contained in this report are those of the author and should not be interpreted as representative of the official policies, either express or implied, of UHCL, RICIS, NASA or the United States Government.

11 February 1992

To: C. J. Melebeck, Barrios Corp.

Subject: Interface Descriptions for GHG Extensions and Functions
for ASV3 and NELS 1.1.

Enclosed is a description document listing the scripts and describing the interface protocols and parameters necessary to run the scripts. GHG will be building to the same document as far as these functions are concerned. If you have any questions, please call David L. Henning at 488-8806 or daveh@ghg.hou.tx.us.

David L. Henning
Systems Engineer
GHG Corporation

Submittal Requirements:

Copies should be provided to the following individuals:

Two Copies to: (Presumably, he will give one to Barrios)

Mark E. Rorvig
Software Technology Branch
Johnson Space Center
NASA

Info: (FAX)

E.T. Dickerson
Level 3 RBSE Program Manager
Research Institute for Computing and Information Systems (RICIS)
University of Houston - Clear Lake

Charles McKay
Chief Scientist RBSE Program
Research Institute for Computing and Information Systems (RICIS)
University of Houston - Clear Lake

Karen Fleming
AdaNET Operations Manager
MountainNet
Delslow, WV

Document Title:

Design Document
ASV3 Extensions/Functions for NELS 1.1

Rev Date 4 March, Initial
DocId SSDD-00005

Original On File with CM
WordPerfect 5.0
File Name: NELS1_1E.NOT

Statement of Capabilities:

This software release conforms to the Interface Control Document, ASV3 Extensions/Functions for NELS 1.1, SSDD-00005, 4 Mar 1992 and the memo, Proposed File Naming Conventions used within the GHG Extensions, 13 Mar 1992 with the following exceptions:

1. Scripts are not provided to support a request for copy to 9 Track 1600 BPI tape in either the TAR or ASCII format.
2. Scripts are not provided to support a request for copy to 14.5x11 hardcopy or 8.5x11 hardcopy.
3. Scripts implementing media copy to IBM or MAC media consist of FTPing files to a PC or MAC in FTP Server mode. Once the files arrive at the PC/Mac they can be transferred to whatever media has been requested as a manual offline process.
4. All references to pathways are implemented relative to \$AUTOLIB_HOME instead of \$ASV3.

All exceptions reflect the lack of appropriate testbed equipment to verify capabilities.

All software has been developed and tested on UHCL computers. The testbed software used BTI code from 18 Mar 1992. The following functions have been verified:

1. The COPY function (Request for Media) has been executed for TAR format 1/4 inch Cartridge tapes and IBM diskettes. GHG did not have access to a MAC with a FTP Server to verify transmission from UHCL to a MAC.
2. The PRINT function has been verified by printing text and Postscript documents on GHG printers from the UHCL computer.
3. The Local Copy has been fully tested.
4. The Download function supports FTP and UUCP only. The target environment was X users who wouldn't be likely to download via Kermit or Zmodem.
5. All help buttons and dialogs which are part of the GHG extensions have been validated as working.
6. The Test Plan update has been fully executed and verified.

The following files reflect either new GHG provided source or GHG updated source. BTI will have to integrate these files into the NELS 1.1 baseline.

C Source	Function
ghg.h	Include for GHG Extensions
ghg.c	Source for X interface GHG Extensions
pipe.h	Include for interface to FTP
pipe.c	Source for interface to FTP
makefile	Updated BTI make file
browser.c	Updated BTI source
help.c	Updated BTI source
oms.c	Updated BTI source
Scripts	Function
media.y	Yacc source for creation of media queues
media.l	Lex source for creation of media queues
copy.y	Yacc source for local copy
copy.l	Lex source for local copy
print.y	Yacc source for print
print.l	Lex source for print
main.c	Common function interface for media/local copy/print
ftp.y	Yacc source for FTP to FTP Server
ftp.l	Lex source for FTP to FTP Server
ftpmain.c	Entry or FTP to FTP Server
ftp.h	Include for FTP to FTP Server
cart.y	Yacc Source for TAR to Cartridge
cart.l	Lex source for TAR to Cartridge
cart.c	Entry for TAR to Cartridge
makefile	Make file for scripts
Help Files	Function
helpindex	Updated BTI file
request.output	main help for output request button
request.output.copy	help for copy function
request.output.download	help for download functions
request.output.ftp	help for ftp form
request.output.local_copy	help for local copy function
request.output.now_or_later	help for session capability
request.output.print	help for print functions
request.output.session	help for exit functions
request.output.uucp	help for uucp form

TABLE OF CONTENTS

SECTION 1.....	3
INTRODUCTION.....	3
SECTION 2.....	4
TEST OBJECTIVES.....	4
SECTION 3.....	5
SRS / TEST PROCEDURE CROSS-REFERENCE MATRIX.....	5
SECTION 4.....	13
TEST IMPLEMENTATION.....	13
4.1 HARDWARE AND SOFTWARE REQUIREMENTS.....	13
4.2 LOCATION AND SCHEDULE.....	13
4.3 PREPARATION OF INPUTS.....	13
SECTION 5.....	14
TEST CASES.....	14
5.1 USER INTERFACE REQUIREMENTS.....	14
5.1.1 Help.....	14
5.1.2 Screen Layout.....	30
5.1.4 Mouse Interface.....	46
5.1.5 Menu Driven Interface.....	47
5.1.6 System Exit.....	47
5.2 USER FUNCTIONS.....	61
5.2.1 Browsing.....	61
5.2.2 Retrieval.....	79
5.2.3 Retrieval Options.....	99
5.2.4 Tools.....	101
5.2.5 Reporting.....	103
5.2.6 User Profile.....	107
5.3 LIBRARIAN FUNCTIONS.....	111
5.3.1 Access Privileges.....	111
5.3.2 Object Classes.....	117
5.3.3 Collections.....	125
5.3.4 Tools.....	130
5.3.5 Objects.....	133
5.3.6 Synonym Table.....	137
5.4 AUDIT.....	138
5.5 OUTPUT REQUEST.....	139

Design Document
ASV3 Extensions/Functions for NELS 1.1

1.0 Background.

This document describes the GHG Functions and Extensions to be added to the NELS 1.1 product. These functions will implement the "Output Request" capability within the Object Browser. The functions will be implemented in two parts. The first part is code to be added to the Object Browser (X Version) to implement menus allowing the user to request that objects be copied to specific media, or that objects be downloaded to the user's system following a specific protocol, or that the object be printed to one of printers attached to the host system. The second part is shell scripts which support the various menu selections. Additional sScripts to support functions within the GHG Shell (X Version) will also be created along with the X Version of the GHG Shell as initial capability for the March 27 prototype. The scripts will be composed of C shell routines that will accept parameters (primarily file pathways). Certain limitations in functionality will be imposed for the March increment. For instance, the E-Mail functions will invoke Mail instead of Oracle Mail since that has yet to be delivered and the NELS invocation will default to the X-Window version instead of the ASCII version.

2.0 Overview.

2.1 GHG C Source Code.

GHG will prepare source code for delivery to NASA which will implement the "Output Request" function within the NELS Object Browser. It will be up to NASA to oversee the integration of this code into other NELS Development efforts. The GHG source code will create a button on the NELS Object Browser labeled "Output Request". Various references to metadata in the following functions should be interpreted as a linear list in the form:

data label: data value

.
.
.

data label: data value

as defined by the class definition of each object.

Design Document
ASV3 Extensions/Functions for NELS 1.1

2.1.1 Activating that button will result in a pulldown menu or an error message window. If no objects had been selected prior to the "Output Request" button activation then the error message window will display the message "You must have selected an object or objects prior to requesting output." If the menu appears, it will list the following selections:

- Copy
- Print
- Download
- Local Copy

2.1.2 The Copy Function.

The Copy Function is intended to allow the user to request that specific objects and their accompanying metadata be transferred to specific media. Selecting the Copy option from the Output Request Menu results in an additional pulldown menu detailing the various devices or media that may be copied to. See Figure 3-1. The user selects an appropriate device causing the following actions to occur:

- a. The userid of the user and the current datetime stamp are used for a file name (e.g. userid.datestamp) which is created under the directory \$ASV3/Customer.Service/device where device is a clear text name similar to those listed in Figure 3-1.

- b. The following data is written to that file:

```
.device
device code
.enddevice
.metadata
metadata text for object 1 for as many lines as it takes
.endmetadata
.pathway
pathway for object 1
.endpath
.
.
.
.metadata
metadata text for object n as many lines as it takes
.endmetadata
.pathway
pathway for object n
.endpath
```

Design Document
ASV3 Extensions/Functions for NELS 1.1

c. As each object is written to the file, a record of its access, datetime of access request, and userid of who accessed the file is written to the history file (an Oracle Table).

The Copy Function is then complete. A supporting Client Services function accesses the files produced and completes the requested transfer of data using GHG provided shell scripts.

Device code and clear text name table:

1	3.5 High Density IBM or PC
2	3.5 Low Density IBM or PC
3	3.5 MAC
4	5.25 High Density IBM or PC
5	5.25 Low Density IBM or PC
6	9 Track 1600 BPI tape (TAR)
7	9 Track 1600 BPI tape (ASCII)
8	1/4 inch Cartridge Tape
9	Hardcopy (14.5 by 11)
10	Hardcopy (8.5 by 11)

Figure 3-1

Design Document
ASV3 Extensions/Functions for NELS 1.1

2.1.3 Print.

Activation of the Print option will cause the display of a pulldown menu listing the available printers. The user will select a printer causing the code to generate a file called \$ASV3/print containing the following data:

```
.device
device code (site dependent, stored in PRINTCAP)
.enddevice
.metadata
metadata text for object 1 for as many lines as it takes
.endmetadata
.pathway
pathway for object 1
.endpath
.metadata
metadata text for object 2 as many lines as it takes
.endmetadata
.pathway
pathway for object 2
.endpath
.
.
.
.metadata
metadata text for object n as many lines as it takes
.endmetadata
.pathway
pathway for object n
.endpath
```

Design Document
ASV3 Extensions/Functions for NELS 1.1

2.1.4 Download.

Activation of the Download option of the Output Request Menu generates a second pulldown menu with the following options:

FTP
KERMIT
X-MODEM
Y-MODEM
Z-MODEM
UUCP

Initially, the only active options will be FTP and UUCP. This is due to the inability to "escape" back to one's home site and activate a similar utility. The ASCII interface will support all of the above options except FTP. Non-available options will be disabled or "grayed out".

2.1.4.1 FTP.

Selection of the FTP option will create a new window with the data fields "Target Address", "Target Userid", and "Target Password". The window will also have buttons for "OK", "Cancel", and "Help". The user must fill in the fields and press "OK", cancel the operation, or press "help" for an explanation. Pressing "OK" will cause a check for empty fields and an appropriate error message. Assuming that the fields are filled in, the function will build temporary files for metadata of the objects selected and the associated pathways of the objects. If the "script file" already exists (explained in a moment), then data is appended. After recording all of the data, it will ask the user if transfer should be started now or postponed. If the user answers "Now" then it will then activate FTP and pass the pathway of the metadata and object pointers for transfer to the target address using the target userid and password. At the end of the FTP function, the temporary files are deleted. If the user chooses to postpone transfer, then the file is left open so that additional data may be appended later. The naming convention of the temporary files for metadata is \$ASV3/object_file_name.metadata. The "script file" will be named \$ASV3/userid.FTP. The format of the "script file" is:

```
pathway of object 1 metadata
pathway of object 1
.
.
.
pathway of object n metadata
pathway of object n
```

Design Document
ASV3 Extensions/Functions for NELS 1.1

2.1.4.2 Kermit.

Kermit will use much the same procedures as described for FTP. The difference is that the user will be able to set a variety of variables used by Kermit to regulate the transfer protocols. The variables and the format of the ASCII sequence to get their values will be addressed at a later date.

2.1.4.3 X-Modem, Y-Modem, Z-Modem.

These protocols are closely related. Setting some values controlling Xmodem causes the use of Y-Modem or Z-Modem. The objects and metadata will use much the same procedures as described for FTP. The difference is that the user will be able to set a variety of variables used by Xmodem to regulate the transfer protocols. The variables and the format of the ASCII sequence to get their values will be addressed at a later date.

2.1.4.4 UUCP.

UUCP is a form of Internet Mail. Selection of this option will display a request for the target or "bang" address. UUCP will use much the same procedures for objects and associated metadata files as described for FTP.

2.1.5 Local Copy.

The local copy is provided as a debug tool as much as anything else. It is only available to librarians and not average users. It will create a file named \$HOME/lcopy.datetime with the following data:

```
.metadata
metadata text for object 1 for as many lines as it takes
.endmetadata
.pathway
pathway for object 1
.endpath
.metadata
metadata text for object 2 as many lines as it takes
.endmetadata
.pathway
pathway for object 2
.endpath
.
.
.
.metadata
metadata text for object n as many lines as it takes
.endmetadata
```

Design Document

ASV3 Extensions/Functions for NELS 1.1

```
.pathway
pathway for object n
.endpath
```

A shell script file will then be called to create metadata files and copies of the associated objects in the user's home directory. The same routines which create metadata files used for FTP (and other) transfers will be used following the same file naming conventions. The shell script will not delete the resulting files (unlike the transfer functions).

2.1.6 Exit Processing.

GHG will provide code to be inserted into the exit processing functions of NELS 1.1 which will query the status of a global variable to determine whether outstanding download requests are pending. If download requests are pending the user will be asked to either start the transmission or cancel the request.

2.2 GHG Shell Scripts.

Scripts will be created to support the following functions:

Script Name	Description
Local_Copy	Local copy to home directory
Local_Print	Local print to site printer
Change_Password	Change the user password
Set_Term	Set the term type
Lan_Copy	Modified version of Local_Copy
Tar_Tape	Transfer a list of files to a 1600 BPI Tape
Ascii_Tape	Transfer a list of files to a 1600 BPI Ascii Tape
Cart_Tape	Transfer a list of files to a 1/4 inch Cartridge Tape
Wide_Print	Print a list of files to 14.5x11 forms
Side_Print	Print a list of files to 8.5x11 forms
Print_Stat	Dump NELS statistics using predefined
Oracle	report

The majority of the scripts are very simple. The only complication is providing the capability for multiple selections. Multiple selections will be handled in a uniform way. Multiple selections will be accumulated by appending metadata directive (.metadata) followed by the metadata for the object, followed by the pathway directive (.pathway) and the full file pathway to an ascii file in the users home directory. NELS will open this file for append every time the user indicates that a file should be marked for download, copy or print. There will be a separate file for download, copy and print. The GHG scripts will use this

Design Document
ASV3 Extensions/Functions for NELS 1.1

file as input (required parameter is filename) and destroy the file upon completion of the function. The media request is a little more complicated because more data is required than just file pathways. The media request will use a set of pseudo directives to separate different types of data needed to detail the files to be transferred and the mailing information.

This approach will provide for cumulative identification of objects as well as allowing the list to be used as a clear text list of instructions if customer service chooses to do the job manually. The File name for this media directive should be userid.datestamp which will allow for storage of multiple requests by the same user if necessary. The ".end" directives are required for any multi-line data values.

The customer service personnel can access these media request files and either use download software from the GHG shell or UNIX utilities to print, TAR, or copy the files to the appropriate devices. The GHG shell will not handle downloads of offline files or files stored on other hosts. Nor will the download handle dumps to tape or print with specific forms. Inclusion of the device code allows for exception handling to detect these types of problems.

3.0 Detailed Interface Description.

All script names are case sensitive.

3.1. Local Copy.

3.1.1 Description.

This routine will copy all files listed within the input file to the user's HOME directory. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be copied because of invalid pathways or lack of permissions.

Script name: Local_Code

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

Design Document
ASV3 Extensions/Functions for NELS 1.1

3.2 Local Print.

3.2.1 Description.

This routine will print all files listed within the input file to the printer requested. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be printed because of invalid pathways, lack of permissions, or bad format.

Script name: Local_Print

Required parameters:

character string, full pathway of file list file.

character string, printcap code for required

printer.

Return code: 0 for success, greater than 0 is error.

3.3 Change the users password.

3.3.1 Description.

This routine will allow the user to specify a new password. The routine will return a code of 0 for successful, 1 unsuccessful.

Script name: Change_Password

Required parameters:

none.

Return code: 0 for success, greater than 0 is error.

3.4 Set Terminal Type.

3.4.1 Description.

This routine will allow the user to specify a different terminal type. The routine will return a code of 0 for successful, 1 unsuccessful.

Script name: Set_Term

Required parameters:

character string, entry code for termcap.

Return code: 0 for success, greater than 0 is error.

Design Document
ASV3 Extensions/Functions for NELS 1.1

3.5 Copy Requested Objects to Media.

3.5.1 Description.

There are really two classes of scripts here. Most of the PC/MAC compatible media is not a device directly connected to the host. This necessitates Customer Service transferring these files to a PC or MAC first. This may be done by executing a variant of the local_copy script against the files stored in the various "device" directories followed by FTP or Kermit as an interactive user. For those devices directly connected to the host, specific scripts can be executed using the files stored in the appropriate directory. This approach allows customer service to both control the sequence of media copy and the priority. The file naming conventions userid.datetime allows customer service determine how old the outstanding requests are by simply executing a directory command from within UNIX. The routines will return a code of 0 for successful, 1 for missing parameters, 2 bad device code, 3 for partial success.

Script name: Lan_Copy

Function: Creates a copy of requested files in the home directory of the Customer Service representative.

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

Script name: Tar_Tape

Function: Creates a copy of requested files on a 1600 BPI tape in TAR format.

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

Script name: Ascii_Tape

Function: Creates a copy of requested files on a 1600 BPI tape in ASCII format.

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

Script name: Cart_Tape

Function: Creates a copy of requested files on a 1/4 inch Cartridge Tape.

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

Script name: Wide_Print

Design Document
ASV3 Extensions/Functions for NELS 1.1

Function: Prints the associated metadata and objects on
14.5 x 11 form.

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

Script name: Side_Print

Function: Prints the associated metadata and objects on
8.5 x 11 form (landscape mode).

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

3.6 Print Current Statistics Report.

3.6.1 Description.

This routine will allow the user to request a predefined report. The routine will return a code of 0 for successful, 1 for missing parameters, 2 bad printer code, 3 for bad report name.

Script name: Print_Stat

Required parameters:

character string, Oracle name for the report to be printed.

character string, printcap code for required printer.

Return code: 0 for success, greater than 0 is error.

4.0 Schedule For Development.

Delivery of Initial NELS 1.1 code to GHG from NASA	28 Feb 1992
---	-------------

Development of GHG code and scripts for GHG Extensions to NELS 1.1	4 Mar 1992 - 16 Mar 1992
--	--------------------------

Unit Testing of GHG code	16 Mar 1992 - 24 Mar 1992
--------------------------	---------------------------

Test Plan Development for GHG Code and Scripts	4 Mar 1992 - 16 Mar 1992
---	--------------------------

March 4, 1992

SSDD-00005

Design Document
ASV3 Extensions/Functions for NELS 1.1

Submission of Code, Scripts 24 Mar 1992
and proposed Test Plan for GHG
Extensions and Scripts

25 Feb 1992

Description of the System Processor Module of the RBSE Shell.

Background:

The System Processor is called from the Command Processor when an external command is detected. An external command is a utility or standalone system resident on the host computer. This convention is what allows "plug compatibility" within the GHG Shell for RBSE. External commands are mapped to an arbitrary character string in the file \$ASV3/Command. This is an ASCII text file with a fixed format. See the "Description of the Command Processor Module of the RBSE Shell" for more information on the format and fields within this file. Changing the mapping within this ASCII file is all that is needed to add new or update old external commands. This particular tool is limited for ASV3 in that it will not support passing parameters to the external command.

Basic Functions:

The module receives a character string as input which is assumed to be a standalone executable on the host and reachable (within the user's path). The routine will return an integer value defined as ERR or OK. ERR is interpreted as a failure (command or system not reachable). OK is interpreted as a normal return.

Initial ASV3 Capabilities:

For ASV 3.0, the capabilities will be as described within basic functions. For ASV 3.1, it is likely that the capability will be expanded to include both parameter passing to the called command and enhanced error processing within the system module itself.

25 Feb 1992

Description of the Startup Module for the RBSE Shell.

Background:

The GHG Shell program will provide the RBSE system administrator the flexibility to both tailor screen that the user sees and add or upgrade standalone subsystems without requiring a recompile of the GHG Shell software. This is accomplished by allowing various ASCII files to drive the display process as well as defining the precise command to screen relationships. The startup module is responsible for attaching to various predetermined files and loading their contents into memory. After successful load, a global pointer (defined by the main module) is pointed to the various memory structures.

Basic Functions:

The Startup module is composed of two main routines; Default and Normal. The Default routine will query the file \$HOME/Default for terminal set up parameters as well as load \$ASV3/Disclaimer into memory. If the user has selected to be configured as an X terminal, the Startup module terminates and returns control to the main program which will call the GHG X-Windows Shell. For other terminal types, the Normal routine is required. It loads files \$ASV3/Command and \$ASV3/Screen. These files map commands to screen and other external functions. They are required to drive the Curses and the Cmdline toolkits.

File Structures:

\$HOME/Default:

```
struct default_file
{
    char Xterm;           /* value of non-zero indicates X user */
    char Curses;          /* value of non-zero indicates Curses
                           user */
    char Attr1[3];        /* System default for prot_attr */
    char Attr2[3];        /* System default for unprot_attr */
    char Attr3[3];        /* System default for hilite_prot */
    char Attr4[3];        /* System default for hilite_unprot */
    char Cmdline;         /* value of non_zero indicates a Cmdline
                           user */
    char termtype[10];    /* character string terminal type for
                           set term */
};
```

\$ASV3/Disclaimer:

The Disclaimer file is composed on normal ASCII text that could be read into a single dimensioned array. The various toolkits would make the decision of where line feeds needed to go based on terminal geometry.

\$ASV3/Command:

```
struct cmd_file
{
    char seq[3];
    char cmd[10];
    char cmdtype;
    char scrn_name[20];
};
```

This is a fixed format, fixed length record. Reading the record in should result in allocating another block of memory as described by the cmd_table structure. The sequence number is used for external sorts and can be ignored. The total number of records read should be stored in MAX_COMMANDS.

\$ASV3/Screen:

```
struct scrn_file
{
    char scrn_name[20];
    char scrn_type[1];
    char type_name[20];
    char start_focus[2];
    char curr_focus[2];
    char prot_attr[3];
    char unprot_attr[3];
    char prot_hilite[3];
    char unprot_hilite[3];
};
```

This is a fixed format, fixed length record. Reading the record in should result in allocating another block of memory as described by the scrn_table structure. The sequence number is used for external sorts and can be ignored. The total number of records read should be stored in MAX_SCREENES.

Internal Memory Structures:

```
struct default_def
{
    int Xterm;        /* value of non-zero indicates X user */
    int Curses;       /* value of non-zero indicates Curses
                        user */
    int Attr1;        /* System default for prot_attr */
};
```

```

int Attr2;      /* System default for unprot_attr */
int Attr3;      /* System default for hilite_prot */
int Attr4;      /* System default for hilite_unprot */
int Cmdline;    /* value of non_zero indicates a Cmdline
                  user */
char *termtype; /* character string terminal type for
                  set term */
};

struct disclaimer
{
    char disclaim_text[MAX_DISC_LEN];
};

struct cmd_table
{
    char *cmd;
    char cmdtype;
    char *sys_scrn;
} defined_commands[MAX_COMMANDS];

struct scrn_table
{
    char *scrn_name;
    char scrn_type;
    char *type_name;
    int start_focus;
    int curr_focus;
    int prot_attr;
    int unprot_attr;
    int prot_hilite;
    int unprot_hilite;
} defined_screen[MAX_SCREEN];

```

Initial ASV3 Capabilities:

Initial ASV3 capabilities are as indicated above.

IDENTIFICATION AND RELEASE DATA RECORD

DOCUMENT / SPECIFICATION NUMBER: MOU-00001

NOMENCLATURE OR DOCUMENT TITLE: Interface Descriptions for GHG
Extensions & Functions for
ASV3 & NELS 1.1

PART NUMBER: REV. LEVEL: Init.

RELEASE NUMBER: 00000000000000004 RELEASE DATE: 02/11/92

CONTRACT NUMBER: RBSE-FD-R&T-006-0
Subcontract No. 044
Project No. RICIS No. SE. 18,
Cooperative Agreement NCC9-16

PROGRAM TITLE: Repository Based Software Engineering (RBSE)
Program (AdaNET)

VENDOR/SUBCONTRACTOR 2Y794 GHG Corporation
CAGE CODE & ADDRESS: 1300 Hercules, Ste 111
Houston, Texas, 77058

ITEM TYPE: H/W _____ DWG. TYPE: _____ DWG. SIZE: _____
SHEET COUNT: _____ DASH NO.: _____
S/W _____ CALL NAME: _____ VERSION: _____

DESCRIPTION (17 LINES MAX): A description document listing the
scripts and describing the interface protocols and parameters
necessary to run the scripts. MOU generated at the request of
Dr. C.W. McKay (RICIS), U of Houston (also a recipient)

USED ON (EFFECTIVITY): AdaNet NEXT HIGHER ASSEMBLY: N/A

DEVELOPER: D. Henning QA REF:

CCB CHAIR: R. C. Frederiksen DATE: 02/11/92

CCB (RELEASE) CLERK: Frederiksen DATE: 02/11/92

GHG Corporation
1300 Hercules, Suite 111
Houston, Texas, 77058

11 February 1992

Dr. C. W. McKay
SERC/HTL @ UHCL
Box 447
2700 Bay Area Blvd.
Houston, Texas, 77058-1068

Dear Sir,

Enclosed is a hard copy of the Interface Descriptions required for GHG Extensions and Functions for ASV3 and NEL5 1.1. This paper was generated at your request.

Should you have any questions or if I can be of further service, please don't hesitate to call at (713) 488-8806. Thank you for your time and effort.

Sincerely,

Robert C. Frederiksen
Configuration/Data Management

Distribution:
Dr. C.W. McKay
GHG CM Files

ORIGINAL PAGE IS
OF POOR QUALITY

GHG Corporation
1300 Hercules, Suite 111
Houston, Texas, 77058

11 February 1992

Mr. C. J. Melebek
Barrios Corporation
1331 Gemeni
Houston, Texas, 77058

Dear Sir,

Enclosed is a hard copy of the Interface Descriptions required for GHG Extensions and Functions for ASV3 and NELS 1.1. This paper was generated at the request of Dr. McKay, who is also a recipient.

Should you have any questions or if I can be of further service, please don't hesitate to call at (713) 488-8806. Thank you for your time and effort.

Sincerely,

Robert C. Frederiksen
Configuration/Data Management

Distribution:
Dr. C.W. McKay
GHG CM Files

11 February 1992

To: C. J. Melebeck, Barrios Corp.

Subject: Interface Descriptions for GHG Extensions and Functions
for ASV3 and NEL5 1.1.

Enclosed is a description document listing the scripts and describing the interface protocols and parameters necessary to run the scripts. GHG will be building to the same document as far as these functions are concerned. If you have any questions, please call David L. Henning at 488-8806 or daveh@ghg.hou.tx.us.

David L. Henning
Systems Engineer
GHG Corporation

February 7, 1992

Basic Capabilities/Interface
ASV3 Extensions/Functions for NELS 1.1

1.0 Background.

Scripts need to be created both as extensions for Nels 1.1 and as initial capability for the March 27 prototype. The scripts will be composed of C shell routines that will accept parameters (primarily file pathways). The required scripts include both the extensions to NELS and preliminary functionality for the GHG Shell as specified by MountainNet. Certain limitations in functionality will be imposed for the March increment. For instance, the E-Mail functions will invoke Mail instead of Oracle Mail since that has yet to be delivered and the NELS invocation will default to the X-Window version instead of the ASCII version.

2.0 Overview.

Scripts will be created to support the following functions:

<u>Script Name</u>	<u>Description</u>
Local_Copy	Local copy to home directory
Local_Print	Local print to site printer
Download_FTP	File transfer to another UNIX host
Download_Kermit	File transfer to PC/Mac
Download_Xmodem	File transfer to PC/Mac using Xmodem
Download_Ymodem	File transfer to PC/Mac using Ymodem
Download_Zmodem	File transfer to PC/Mac using Zmodem
Download_UUCP	mail files via UUCP to another UNIX host
Change_Password	Change the user password
Set_Term	Set the term type
Req_Media	Transfer a list of files to requested media
Print_Stat	Dump NELS statistics using predefined Oracle report

The majority of the scripts are very simple. The only complication is providing the capability for multiple selections. Multiple selections will be handled in a uniform way. Multiple selections will be accumulated by appending metadata directive (.metadata) followed by the metadata for the object, followed by the pathway directive (.pathway) and the full file pathway to an ascii file in the users home directory. NELS will open this file for append every time the user indicates that a file should be marked for download, copy or print. There will be a separate file for download, copy and print. The GHG scripts will use this file as input (required parameter is filename) and destroy the file upon completion of the function. The media request is a little more complicated because more data is required than just file pathways. The media request will use a set of pseudo directives to separate different types of data needed to detail the files to be transferred and the mailing information.

February 7, 1992

Basic Capabilities/Interface
ASV3 Extensions/Functions for NELS 1.1

An ascii file in the following format should be created:

```
.addressee
company or individual name
data can be multi-line
.endaddressee
.address
full mailing address for
as many lines as it takes
.endaddress
.attention
use a blank line if no entry is required here
.endattention
.special instructions
use a blank line if no entry is required here
.endspecial
.device
device code and clear text equivalent (see figure 1-1)
.metadata
metadata text for file 1 for as many lines as it takes
.endmetadata
.pathway
pathway for file 1
.metadata
metadata text for for file 2 as many lines as it takes
.endmetadata
.pathway
pathway for file 2
.
.
.
.metadata
metadata text for for file n as many lines as it takes
.endmetadata
.pathway
pathway for file n
```

This approach will provide for cumulative identification of objects as well as allowing the list to be used as a clear text list of instructions if customer service chooses to do the job manually. The File name for this media directive should be userid.datestamp which will allow for storage of multiple requests by the same user if necessary. The ".end" directives are required for any multi-line data values.

February 7, 1992

Basic Capabilities/Interface
ASV3 Extensions/Functions for NELS 1.1

Device code and clear text name table:

1	3.5 High Density IBM
2	3.5 Low Density IBM
3	3.5 MAC
4	5.25 High Density IBM
5	5.25 Low Density IBM
6	9 Track 1600 BPI tape (TAR)
7	9 Track 1600 BPI tape (ASCII)
8	1/4 inch Cartridge Tape
9	Hardcopy (14.5 by 11)
10	Hardcopy (8.5 by 11)

Figure 2-1

The customer service personnel can access these media request files and either use download software from the GHG shell or UNIX utilities to print, TAR, or copy the files to the appropriate devices. The GHG shell will not handle downloads of offline files or files stored on other hosts. Nor will the download handle dumps to tape or print with specific forms. Inclusion of the device code allows for exception handling to detect these types of problems.

3.0 Detailed Interface Description.

All script names are case sensitive.

3.1. Local Copy.

3.1.1 Description.

This routine will copy all files listed within the input file to the user's HOME directory. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be copied because of invalid pathways or lack of permissions.

Script name: Local_Code

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

3.2 Local Print.

February 7, 1992

Basic Capabilities/Interface
ASV3 Extensions/Functions for NELS 1.1

3.2.1 Description.

This routine will print all files listed within the input file to the printer requested. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be printed because of invalid pathways, lack of permissions, or bad format.

Script name: Local_Print

Required parameters:

character string, full pathway of file list file.

character string, printcap code for required printer.

Return code: 0 for success, greater than 0 is error.

3.3 Download via FTP.

3.3.1 Description.

This routine will transfer all files listed within the input file to the host address requested. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be transferred because of invalid pathways, lack of permissions, or bad format.

Script name: Download_FTP

Required parameters:

character string, full pathway of file list file.

character string, host address.

Return code: 0 for success, greater than 0 is error.

3.4 Download via Xmodem.

3.4.1 Description.

This routine will transfer all files listed within the input file to the terminal via the protocol requested. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be transferred because of invalid pathways, lack of permissions, or bad format.

Script name: Download_Xmodem

Required parameters:

character string, full pathway of file list file.

character string, host address.

Return code: 0 for success, greater than 0 is error.

February 7, 1992

Basic Capabilities/Interface
ASV3 Extensions/Functions for NELLS 1.1

3.5 Download via Ymodem.

3.5.1 Description.

This routine will transfer all files listed within the input file to the terminal via the protocol requested. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be transferred because of invalid pathways, lack of permissions, or bad format.

Script name: Download_Ymodem

Required parameters:

character string, full pathway of file list file.

character string, host address.

Return code: 0 for success, greater than 0 is error.

3.6 Download via Zmodem.

3.6.1 Description.

This routine will transfer all files listed within the input file to the terminal via the protocol requested. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be transferred because of invalid pathways, lack of permissions, or bad format.

Script name: Download_Zmodem

Required parameters:

character string, full pathway of file list file.

character string, host address.

Return code: 0 for success, greater than 0 is error.

3.7 Download via UUCP.

3.7.1 Description.

This routine will transfer all files listed within the input file to the host address requested. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be transferred because of invalid pathways, lack of permissions, or bad format.

Script name: Download_UUCP

Required parameters:

character string, full pathway of file list file.

character string, host address.

Return code: 0 for success, greater than 0 is error.

February 7, 1992

Basic Capabilities/Interface
ASV3 Extensions/Functions for NELS 1.1

3.8 Change the users password.

3.8.1 Description.

This routine will allow the user to specify a new password. The routine will return a code of 0 for successful, 1 unsuccessful.

Script name: Change_Password

Required parameters:

none.

Return code: 0 for success, greater than 0 is error.

3.9 Set Terminal Type.

3.9.1 Description.

This routine will allow the user to specify a different terminal type. The routine will return a code of 0 for successful, 1 unsuccessful.

Script name: Set_Term

Required parameters:

character string, entry code for termcap.

Return code: 0 for success, greater than 0 is error.

3.10 Copy Requested Objects to Media.

3.10.1 Description.

This routine will allow the user to request a copy of objects and their associated metadata to different types of media. The routine will return a code of 0 for successful, 1 for missing parameters, 2 bad device code, 3 for partial success.

Script name: Req_Media

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

February 7, 1992

Basic Capabilities/Interface
ASV3 Extensions/Functions for NELs 1.1

3.11 Print Current Statistics Report.

3.11.1 Description.

This routine will allow the user to request a predefined report. The routine will return a code of 0 for successful, 1 for missing parameters, 2 bad printer code, 3 for bad report name.

Script name: Print_Stat

Required parameters:

character string, Oracle name for the report to be printed.

character string, printcap code for required printer.

Return code: 0 for success, greater than 0 is error.

APPENDIX F
DELIVERABLES COVER SHEET

RB.05 GHG Corporation
January 1, 1992

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW # 2

Title of Task: Design document for the exit processor of the GHG shell

Subcontractor: GHG Corporation

Cooperative Agreement No. NCC-9-16

Principal Investigator: E.T. DICKERSON

NASA Technical Monitor: E. FRIDGE

Type of Report: _____

Period Covered by Report _____

Due Date: 02/25/92

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

The Command Processor Module will match text string commands against an ascii definition file to determine which processes to call. This will apply only to the ASCII interface since X is a more complex environment to implement this capability into. This provides the capability of adding or replacing major subsystems to ASV3 without a redesign of the baSic interfaces.

The Screen Manager Module will provide a linkage between submitted commands and resulting menus of choices. It too will function within the confines of the ASCII interface. Look and Feel of the resultant menus is dependent on the extent that the Curses or Command Line Toolkit support tailoring. ASV3 will have a very limited Toolkit capability because of budget and time constraints.

The System Processor Module provides a mechanism where the ASCII interface can be integrated with new stand-alone UNIX based systems. This is similiar to the tool interface within NELS.

The Startup Module will load memory structures through execution of routines within the Default and Normal Modules to control the ASV3 system configuration. User preferences, various ASCII configuration files and definition files for commands, screens, tools, etc. are loaded via the Startup Module. This applies to the ASCII interface only for ASV 3.0.

The Exit Processor assures that any temporary files are purged and user configuration changes are saved. This applies to the ASCII interface only for ASV 3.0.

The Curses Baseed Toolkit is a collection of routines which provide a character based window like interface. The scope of implementation of ASV3 has been reduced to the point where this toolkit is unnecessary.

The Command Line Toolkit is similiar to the Curses Toolkit except that the screen geometry can not be defined and no cursor addressing is supported. The same configuration and definition files support both the Curses and the Command Line Toolkits. The scope of implementation of ASV3 has been reduced to the point where this toolkit is unnecessary.

25 Feb 1992

Description of the Exit Processor of the RBSE Shell.

Background:

The main module will have a number of internal tables allocated within memory. These will have to be released. Any open files will have to be closed. Any child processes will have to be killed. The typical remote user of the RBSE system will have the ASV3 software listed as his/her shell. After a graceful wrapup within the Exit Processor, the remote user will automatically disconnect, providing some small level of system security.

Basic Functions:

The following structures may well be in memory, requiring release:

```
struct form_file
{
    char *form_name;
    int field_seq;
    char *field_label;
    int field_label_row;
    int field_label_col;
    int field_row;
    int field_col;
    int field_length;
    int field_type;
    char *field_contents;
    char *field_choice;
} form_descr[MAX_FORM_SIZE];

struct context_help
{
    char *form_menu_name;
    int field_line_seq;
    int help_seq;
    char *help_text;
} form_menu_help[MAX_HELP_SIZE];

struct menu_tab
{
    char *menu_name;
    int upper_left_row;
    int upper_left_col;
    int type_border;
    int menu_attr;
    int bord_attr;
    int bar_attr;
    int menu_width;
```

```

    int menu_length;
    int menu_start;
    char *menutext[MAX_MENU_LINES];
};

```

```

struct cmd_table
{
    char *cmd;
    char cmdtype;
    char *sys_scrn;
} defined_commands[MAX_COMMANDS];

```

```

struct scrn_table
{
    char *scrn_name;
    char scrn_type;
    char *type_name;
    int start_focus;
    int curr_focus;
    int prot_attr;
    int unprot_attr;
    int prot_hilite;
    int unprot_hilite;
} defined_screen[MAX_SCREEN];

```

```

struct choice_tab
{
    char choice_code[2];
    char *cmd_equiv;
} defined_choices[MAX_CHOICES];

```

Additionally, the following files should be closed if still open:

```

$HOME/user_defaults
$ASV3/disclaimer
$ASV3/Command
$ASV3/Screen

```

The Curses and Cmdline toolkits use various files under the following directories:

```

$ASV3/Forms
$ASV3/Menus
$ASV3/Helps
$ASV3/Choices

```

which they are responsible for closing.

Process-ids spawned from the System Processor should be checked if any are still present in the process stack.

Initial ASV3 Capabilities:

Initial ASV3 functions are as described above.

24 Feb 1992

Description of the Command Processor Module of the RBSE Shell.

Background:

The RBSE Program requires both an interim set of capabilities and a framework for evolving new and more powerful versions of the original capabilities as well as adding completely new functions. An additional constraint (optional but very desirable) is to separate the user interface from the control mechanisms within the system. The ideal end result would allow new functions to be added, existing functions to be replaced, and new user interfaces defined without requiring a system redesign or even a system compile. There are some tradeoffs for this flexibility. It is highly likely that a system with the flexibility described would not be as efficient in either processor or file utilization as one with less flexibility.

Basic Functions:

The command processor consists of a routine which receives an initial parameter as a sort of seed (in this case the seed is the command "main"). The command is matched against an internal table which contains the following information:

```
struct {
    char *cmd;      /* the command string, e.g. "main" */
    char cmdtype;   /* describes the following parameter as either a
                    screen name or a system call string */
    char *sys_scrn /* screen name or system command string */
};
```

If the command type indicates that the following table element is a screen name then the screen manager is called with an input parameter of the matching screen name. If the command type indicates a system command then the system processor is called with an input parameter of system command. In both cases, input parameters are character strings. In the case of the screen manager, the expected return is a character string. The returned character string are expected to be new commands so the table look up for a matching command definition is executed. This loop continues until the command "off" is received. In the case of the System Processor, the expected return is an interger defined as ERR or OK. A return of ERR should display a message to the user that the command failed. The Command Processor should simulate a return of the command "main" and loop.

Initial ASV3 capabilities:

The ASV3 implementation is very minimal at this point. The only functions defined are the external systems; library manager and e-mail, and the internal screens for the main menu, supporting help

screens and a rudimentary terminal setup screen. All data describing these screen and external commands will be resident in the file \$ASV3/Commands. The file will be a series of text lines having the following structure:

Cols	Element Name	Description
01-03	seq number	Can be used to order the file or imply priority
04-13	command	command literal returned by screen manager or system manager
14	cmdtype	following element is either screen name or system command. This field can assume one of two values [r or y], Screen name or System Command respectively. Future implementations may allow more types.
15-34	action_screen	literal to be used as an input parameters when calling screen manager or system manager.

All elements are required. The file is loaded into memory as a result of normal startup. The resulting memory structure (see preceeding paragraph) is stored and referenced by a global pointer. The structure will be terminated with NULL values in each element.

Initial Values for the ASV3 Command Table:

The following value will be present in the file \$ASV3/Command for the initial installation:

001main	rmain_menu
002alms	sasciilib
003xlms	soms
004email	sMail
005xemail	sxmh
006util	rutility_menu
007mmhelp	rmain_help
008ttset	rsetterm_type
009tthelp	rsetterm_help
010uthelp	rutil_help
011help	rabout

100

—

Figure 6

Initial Values for the ASV3 Screen Table:

The following values will be present in the initial installation:

001main_menu	fmain_menu
002utility_menu	futility_menu
003main_help	fmain_help
004setterm_type	fsetterm_type
005setterm_help	fsetterm_help
006util_help	futil_help
007about	fabout

};

System defaults referenced in the remarks default startup routine. The character st *type_name is passed to either forms or me in ASV 3.0, the only calls will be to the call, 'the screen manager will determine w Cmdline mode. There are two versions of t Curses and one for Cmdline. The form and screens and get the user input. User input manager as a command string and optional manager. (The data manager will be imple capabilities upgrade.)

Initial ASV3 Capabilities:

The ASV3 implementation is very minimal a forms defined have a special variable type simulates a traditional menu as opposed to provides a scrolling menu with bar. Help format form. The other form field types with the menu manager but testing of the not be as rigorous as the initial screens holding the screen descriptions will have

Cols	Element Name	Description
01-03	seq_no	Can be used to on priority.
4-23	scrn_name	name of the file
24	scrn_type	'f' for form or
25-44	type_name	name of the assoc description file
45-46	start_focus	starting menu lin is 01-99.
47-48	curr_focus	current menu line is 01-99.
49-51	prot_attr	display attribut 0-255
52-54	unprot_attr	display attribut 0-255
55-57	hilite_prot	display attribut range is 0-255
58-60	hilite_unprot	display attribut range is 0-255

All elements are required. The file is result of the normal startup routine. T (see preceeding paragraph) is stored and pointer. Attributes with the value zero The structure will be terminated with N

February 7, 1992

Curses Toolbox for ASV3

Background:

This toolkit was developed to support the RBSE program. The primary user of this software is a PC user using a VT100 emulator over a dial-in modem. The goal of the ASV3.x portion of the program was to upgrade the user interface from a command-line style interface to a more modern style of interface. Since the platform or server is a UNIX based system, Curses was selected as the primary screen handling library. Additional interfaces implemented for the ASV3.x upgrade included X-Windows using Motif 1.1. This toolkit will not attempt to emulate X-Window capabilities with its user controlled geometry and font capabilities but will provide some predefined windows like objects which combine lower level Curses routines to a higher level construct similiar to a widget.

The particular model for this interface was HyperPad 2.0, a IBM PC based workalike for HyperCard. Although both HyperPad and HyperCard are event driven interfaces in the same manner as X-Windows, this toolkit will not support event driven capabilities. It simply moves the user from a line oriented environment to a screen oriented environment. Since it is based on Curses, the additional communication traffic will be kept to a minimal level. It is likely that there will still be additional communications overhead using these routines as compared to line oriented routines.

The HyperPad environment was used for prototyping. Various "widget" like constructs are present. For example, a text window which scrolls, allows insert/delete, cut/paste, and follows Microsoft editing conventions (as present in Word and various Microsoft editors). Another artifact is a popup menu which implements a scrollbar, hotkeys and supports multiple selections. Finally, there is a message box which contains one to three buttons. Various components are also present; fields (special cases of the text window artifact), buttons, and check boxes.

The toolkit will be broken down into the following categories:

- Window Utilities
- Field Manager
- Menu Manager
- Form Manager
- Help Manager
- Directory Manager

These modules (initial design language will be ANSI C) will be quite interelated. The Curses subroutines will follow System V conventions because of support for color attributes. Migration to the UNIX platform will require

February 7, 1992

Curses Toolbox for ASV3

insertion of TERM logic to enable TERMCAP or TERMINFO capabilities. These are very minor differences and the result should be a "portable" character based windowing library which can be adapted to a variety of future projects. Potential modifications required for ADA conversion have not been part of the design criteria. This toolkit uses the defined constants MAX_LINES and MAX_LEN. These could be implemented as MAX_LINES = ROWS and MAX_LEN = COLS under UNIX.

General Description of Routines:

Windowing Utilities:

Background:

Windowing utilities provide the most direct use of Curses in that they define areas of the screen as scrolling windows which have a uniform style of bordering and titles. The routines controlling text wrap are also located in this module.

The following functions are contained in this module;

make_window

Prototype:

```
WINDOW *make_window(int totlines, int totcols, \
    int upperleft_absrow, int upperleft_abcscol, char *title, \
    int type_border, int window_attr, int border_attr)
```

Synopsis:

This is a convenience routine that will create a window using newwin() and supplied parameters for positioning and attributes. The window can be bordered with a separate border attribute using either single or double line borders. Parameters required are the total lines in the window (including the border), the total width of the window (including the border), the absolute coordinates of the upper left corner of the window, a window title (centered within the border), the type of border (SINGLE, DOUBLE, or NONE), the window attribute, and the border attribute. The routine will return the window pointer if successful and a null pointer if not.

Example:

```
/* BUILD A WINDOW */
main_menu = make_window(10,55,2,0,"Main Window",DOUBLE, \
    A_NORMAL,A_BOLD);
if (main_menu == (WINDOW *) NULL)
{
```

February 7, 1992

Curses Toolbox for ASV3

```
perror("Can not Create new window");
exit(50);
```

```
}
```

display_with_wrap

Prototype:

```
int display_with_wrap(WINDOW *win, char *string, int row)
```

Synopsis:

This routine will display strings to the window wrapping at word boundaries. If the row parameter is zero, the current window cursor position will be used as a starting point otherwise the string will be displayed starting at the left most position in the named row. If the string exceeds the remaining space available in the window, the window will scroll (if `scrollok(TRUE)`) or return `ERR` (if `srollok(FALSE)`). Required inputs are window pointer, string pointer and row (relative to the top of the window).

Example:

```
/* DISPLAY WRAPPED TEXT IN WINDOW */
display_with_wrap(another_window, \
    "This is an example of a string being written to the window", \
    3);
wnoutrefresh(another_window); /* update virtual area */
doupdate();                  /* update screen */
```

display_with_chop

Prototype:

```
int display_with_chop(WINDOW *win, char *string, \
    int row, int max_strng)
```

Synopsis:

This routine will display strings to the window truncating at word boundaries. If the row parameter is zero, the current window cursor position will be used as a starting point otherwise the string will be displayed starting at the left most position in the named row. If the string exceeds the remaining space available in the window, the routine will return `ERR`. Required inputs are window pointer, string pointer and row (relative to the top of the window).

Example:

February 7, 1992

Curses Toolbox for ASV3

```
/* DISPLAY CHOPPED TEXT IN WINDOW */
display_with_chop(another_window, \
    "This is an example of a string being written to the window", \
    3);
wnoutrefresh(another_window); /* update virtual area */
doupdate(); /* update screen */
```

display_with_align

Prototype:

```
int win_center(WINDOW *win, char *string, int row)
```

Synopsis:

This routine will center strings within the window. Parameters required are the window pointer, string pointer and the target row of the window (as opposed to the absolute row of the screen) Row 0 is the top border row. The row will be compared to the bottom of the window to assure that a legal value is passed. Strings that exceed the width of the window will be truncated. The routine will return the value OK or ERR as defined by the CURSES.H file. The string is not echoed to the screen and requires a "doupdate()" to display.

Example:

```
win_center(win, wind_title, 0);
```

dim_window

Prototype:

```
void dim_window(WINDOW *win)
```

Synopsis:

This routine resets the attributes of a window as DIM, and redisplay the window. No values are returned. Only the window pointer is required as input.

Example:

```
dim_window(main_menu);
```

erase_window

Prototype:

```
void window_erase(WINDOW *win)
```

February 7, 1992

Curses Toolbox for ASV3

Synopsis:

This routine will erase the window from the screen. No values are returned and only the window pointer is required as input.

Example:

```
window_erase(another_window);
```

set_focus

Prototype:

```
void set_focus(WINDOW *win)
```

Synopsis:

This routine will cause the window to be redrawn on the screen. If other windows had covered portions up they are covered up as if the window was brought from the background to the foreground and the previous windows are put into the background. No values are returned and only the window pointer is required as input.

Example:

```
set_focus(main_menu);
```

Field Manager:

The following functions are contained in this module;

display_time

Prototype:

```
int display_time(char *field_in, char *field_out)
```

Synopsis:

This routine will format time fields in the format of 99:99 am/pm. The routine returns OK or ERR.

display_date

Prototype:

```
int display_date(char *field_in, char *field_out, int type_format)
```

Synopsis:

February 7, 1992

Curses Toolbox for ASV3

This routine will format date fields in the format of either Day of the Week, mm/dd/yy or dd mon yy or mm/dd/yy. The routine returns OK or ERR. Type_format is either 1, 2 or 3 respectively.

display_number

Prototype:

```
int display_number(char *field_in, char *field_out, \
                  int type_format, int field_length)
```

Synopsis:

This routine will format number fields in one of the following ways:

- | | | |
|---|---------|---|
| 1 | \$99.99 | fixed precision, leading dollar sign, no
intervening spaces, must be two place integer portion, if
leading zeroes required then add them. Add commas for
integer portions over three characters (i.e. \$99,999.00) |
| 2 | 9,999 | integer with commas |
| 3 | 99.99 | fixed precision. The output fields will be right justified
and space filled up to the limit of the field length. The
routine will return OK or ERR. |

The type_format is 1, 2, or 3 respectively.

display_text

Prototype:

```
int display_text(char *field_in, char *field_out, int field_length)
```

Synopsis:

This routine will parse text so that the input field is written to the output field left justified (leading spaces eliminated, all non-printing characters eliminated) and truncated to fit into the output field up to the length specified in the field length. If the input field needs to be truncated then it must be truncated at word boundaries and the pointer to the input field adjusted to the beginning of the next word following the truncation. All non-printing characters must be removed from the input string. The routine will return ERR, OK, and OVRFLO.

edit_time

Prototype:

```
int edit_time(char *field_in, char *field_out, char *err_msg)
```

February 7, 1992

Curses Toolbox for ASV3

Synopsis:

This routine will examine the incoming field for conformance to 99:99 am/pm. The routine will return the following values: OK, HOURS, MIN, D_N. Any error message will be stored in the err_msg parameters. Error messages are:

- Hours out of range, must be 1 - 12
- Minutes out of range, must be 0 - 59
- Must indicate am or pm, format is 99:99 am or 99:99 pm

edit_date

Prototype:

int display_date(char *field_in, int type_format, char *err_msg)

Synopsis:

The routine will edit the input field for conformance to the following rules:

type_format 1, Day of the Week, mm/dd/yy

- Day of the week may be any case and abbreviated.
- Abbreviations will translated to proper day name.
- Legal abbreviations/translations:

- mon Monday
- tue Tuesday
- wed Wednesday
- thu Thursday
- fri Friday
- sat Saturday
- sun Sunday

- day values must be checked for range conformance relative to month, month must be between 1 to 12, year may be between 50 - 49 (1950 - 2049)

return values are OK, WEEK, DAY, MONTH, YEAR where anything other than OK is an error. Error messages are:

- Day of the week misspelled, use three character abbreviations.

- Day of the week abbreviation misspelled, format is thu 3/29/92.

- Day out of range for month.

- Month must be 1 - 12.

- Year is between 50 - 49

type_format 2, dd mon yy

February 7, 1992

Curses Toolbox for ASV3

day values must be checked for range conformance relative to month. Month must be an abbreviation (case doesn't matter). Legal abbreviations are: jan, feb, mar, apr, may, jun jul, aug, sep, oct, nov, dec. Year may be between 50 - 49 (1950 - 2049).

return values are OK, DAY, MONTH, YEAR where anything other than OK is an error. Error messages are:
Month abbreviation misspelled, format is 29 mar 92
Day out of range for month
Year is between 50 - 49

type_format 3, dd/mm/yy.

Day values must be checked for range conformance relative to month. Month must be between 1 to 12. Year may be between 50 - 49 (1950 - 2049).

return values are OK, DAY, MONTH, YEAR where anything other than OK is an error. Error messages are:
Month abbreviation misspelled, format is 29 mar 92.
Day out of range for month.
Month must be 1 - 12.
Year is between 50 - 49.

edit_number

Prototype:

```
int display_number(char *field_in, int type_format, \
                  int field_length, char *err_msg)
```

Synopsis:

The routine will edit the input field for conformance to the following rules:

Type_format 1, \$99.99 fixed precision, leading dollar sign. There must be a one leading dollar sign. There must be one decimal point. There must be two decimal places after the decimal point. The number can not exceed (field_length) characters.

Return values are OK, DEC, PLACE, SIGN, LENGTH. Error messages will be:
There must be a one leading dollar sign.
There must be one decimal point.

February 7, 1992

Curses Toolbox for ASV3

There must be two decimal places after the decimal point.

The number can exceed %s characters, field_length.

Type_format 2, 9,999 Integer with commas. Commas, if used must occur after the thousands place decimal points are not allowed. The number can not exceed (field_length) characters, negative numbers are not allowed, dollar signs not allowed,

Return values are OK, LENGTH, DEC, SIGN, COMMA. Error messages will be:

No decimal points allowed.

Number is too long to fit.

Number can not be negative or have dollar sign .

Commas if used must occur after the thousands place.

Type_format 3,99.99 Fixed precision. Commas if used must occur after the thousands place. Can't have more than two characters after decimal point. The number can not exceed (field_length) characters, negative numbers are not allowed, dollar signs not allowed.

Return values are OK, LENGTH, DEC, SIGN, COMMA. Error messages will be:

Too many decimal places.

Number is too long to fit.

Number can not be negative or have dollar sign.

Commas if used must occur after the thousands place.

edit_text

At this point no text edits have been specified. Simply call the display_text routine to "clean" the field.

clear_field

Prototype:

```
int clear_field(char *field_in, int type_field,
                int field_length)
```

Synopsis:

February 7, 1992

Curses Toolbox for ASV3

This routine will clear out fields based on the type of field specified in the type_field parameter. The input field will "cleared (spaced out)" and reset to the values indicated for each type.

```
TIME      00:00 am
DATE1     Sunday, 12/31/50
DATE2     31 DEC 50
DATE3     / /
INTEGER   0 (right justified)
FIXED     0.00 (right justified)
DOLLAR    $00.00 (right justified)
TEXT
```

Forms Manager:

Background:

The Forms Manager regulates movement between buttons and fields associated with specific forms as well as display forms. Each form will have a name and a definition stored in an ASCII file somewhere. The form definition file will have the following format:

```
struct form_file
{
    char form_name[20];
    char field_seq[3];
    char field_label[MAX_LEN];
    char field_label_row[2];
    char field_label_col[2];
    char field_row[2];
    char field_col[2];
    char field_length[3];
    char field_type[2];
    char field_contents[MAX_LEN];
    char field_legal_values[MAX_LINES, 10];
};
```

This is a fixed length record with fixed column definitions. The contents of this file is read into a memory structure and ordered by the field_seq value. The memory structure is as follows:

```
typedef struct
{
    char *form_name;
```

February 7, 1992

Curses Toolbox for ASV3

```
int field_seq;
char *field_label;
int field_label_row;
int field_label_col;
int field_row;
int field_col;
int field_length;
int field_type;
char *field_contents;
char **field_legal_values;
) FORMS;
```

Valid field_types are TIME, DATE1, DATE2, DATE3, INTEGER, FIXED, DOLLAR, TEXT, BUTTON, CHECK_BOX and CHOICE. Most of these are defined in the Field Manager section. A button is defined as a field which can not be written to but has a display value defined in field_contents as well as being surrounded by a box. Box coordinates are field_row-1, field_col-1 with totcols = field_length+2 and totrows = 3. Check boxes have a label. The value in the field_contents maybe the constant string of TRUE or FALSE. The field_length is always 1, the box coordinates are field_row-1, field_col-1 with totcols = 3 and totrows = 3. The value of TRUE in the field_contents result in a literal of 'X' being displayed at the field coordinates. FALSE displays a blank.

Fields will follow the same basic conventions as far as keyboard input is concerned. TAB and SHFT_TAB navigates from field to field. RETURN/ENTER activates buttons, toggles checkboxes, and causes the associated edits to be performed. Entering META<n> where n is some number will cause the focus to shift to the field with that field_seq value. META maybe defined differently for various terminals. Pressing the right or left arrow should advance or backup the cursor. Keys should be defined to support the following functions; end_of_field, beginning_of_field, toggle_insert, delete_character, begin_block, end_block, cut_block, copy_block, paste_block and of course help. There should be provision for a separate backspace and a separate delete key. A call to find_help_entry will be made every time the help key is pressed.

Fields which have "focus" will have the display attribute of UNPROTECT (normally defined as REVERSE or UNDERLINE on monochrome displays). Field labels (and the rest of the screen) will have the display attribute of PROTECT (usually the attribute NORMAL). Loading a form will not automatically load a help structure. Any help structures must be loaded prior to calling get_field and should be done after calling get_form.

File naming conventions:

February 7, 1992

Curses Toolbox for ASV3

To assist in finding the appropriate form files, the following file naming conventions will be observed. The pathway to the form file will consist of the default directory for the ASV3 software (\$ASV3) with '/forms/' appended followed by the value stored in the form_menu_name portion of the record.

The following functions are contained in this module;

goto_field

Prototype:

```
int goto_field(int fieldnbr)
```

Synopsis:

This routine will search the form table in memory for the field_seq = fieldnbr. Once found the field_contents will be redisplayed with the attribute UNPROTECT. If focus was changed from a previous field, that field's contents have to be redisplayed with attribute PROTECT. The routine will return OK or ERR. An error would be a fieldnbr that didn't exist.

display_form

Prototype:

```
int display_form(char *form_name, char *form_array[])
```

Synopsis:

This routine will clear the screen and display the form contained in the specified form array. Refer to the background section of Form Manager for details on the form array structures. The routine will return OK or any error value returned by Curses along with a message indicating which field caused the problem (use perror()).

get_form

Prototype:

```
int get_form(char *filename, char *form_array[])
```

Synopsis:

This routine will find the pathway/filename and load the file contents into the designated array. Refer to the background section of the Form Manager for file and array formats. The routine will return MEM, ERR, or OK. MEM will signify that memory is exhausted, ERR signifies that the pathway was not found.

February 7, 1992

Curses Toolbox for ASV3

get_field

Prototype:

```
int get_field(int fieldnbr)
```

Synopsis:

This routine will scan the keyboard for inputs and act based on the type of field having focus. Refer to the discussion in the background portion of Form Manager for global conventions. Specific field rules are as follows:

TIME

Valid characters entered are 0-9, aApPmM, : entering a colon should advance the cursor to position 3. If only one number precedes the colon then prepend a leading zero on the field before sending it to edit. If a return is encountered after the colon then append "00 am" for hours from 8-12 and "00 pm" for hours 1-7.

DATE1

If the user presses META<D> then the current date should be put into the field using DATE1 rules.

DATE2

If the user presses META<D> then the current date should be put into the field using DATE2 rules.

DATE3

If the user presses META<D> then the current date should be put into the field using DATE3 rules. Valid characters are 0-9 /. If the user enters a slash and the preceding numeric portion is only 1 character prepend a leading zero on that portion. If the user presses ENTER/RETURN and the last portion is only one character then prepend a zero to that portion. If a ENTER/RETURN follows the second slash append the current year. A year of 00 is to be interpreted as the year 2000.

INTEGER

No special rules.

FIXED

No special rules.

DOLLAR

February 7, 1992

Curses Toolbox for ASV3

No special rules.

TEXT

No special rules.

BUTTON

No special rules.

CHECK_BOX

No special rules.

Help Manager:

Background:

The Help Manager and the Menu Manager have much in common. The help capability itself consists of two different constructs, the message box and the scrolling locked text window. The message box is a window (bordered) with a title, centered text and one or more buttons in the bottom third of the window. The minimum depth of a message box is 9 lines or 8 lines plus the number of lines in the message. The help window is a text window which doesn't allow input but can scroll up and down. The window should have a top title and bottom instructions. Typical key commands recognized would be scroll_up, scroll_down, and ESC. The contents of help windows are loaded from disk (ASCII files). The helps can be made contextual by prepending the form/menu name and the field/line number to the help. A positioning routine could then find the appropriate starting point.

File Structure for contextual help records (AGAIN NOTE THE FIXED COLUMN DEFINITIONS):

```
struct context_help
{
    char form_menu_name[20];
    char field_line_seq[3];
    char help_seq[3];
    char help_text[MAX_LEN];
};
```

Memory structure for contextual helps:

```
typedef struct
{
    char *form_menu_name;
```

February 7, 1992

Curses Toolbox for ASV3

```
    int field_line_seq;  
    int help_seq;  
    char *help_text;  
} HELPS;
```

File naming conventions:

To assist in finding the appropriate help files, the following file naming conventions will be observed. The pathway to the help file will consist of the default directory for the ASV3 software (\$ASV3) with '/helps/' appended followed by the value stored in the form_menu_name portion of the record.

The following functions are contained in this module;

create_table_space

Background:

This is a generalized routine for loading ASCII data into a memory structure for later display. This is for whatever is needed besides helps and menus.

```
{  
typedef struct  
{  
    char *string[MAX_TABLE];  
    int max_str_len;  
    int nbr_lines;  
} TABLE;
```

Prototype:

```
TABLE *create_table_space(char *src)
```

Synopsis:

This routine will accept a path string, open and read the corresponding text file into memory saving pointers to the strings into a pointer table for future reference. Also saved is the length of the longest string and the total number of strings read. A NULL value returned denotes an error.

create_help_entry

Prototype:

```
HELPS *create_help_entry(char *src)
```

Synopsis:

February 7, 1992

Curses Toolbox for ASV3

This routine will accept a path string, open and read the corresponding contextual help file into a help memory structure saving pointers to the strings into a pointer table for future reference. A NULL value returned denotes an error.

find_help_entry

Prototype:

```
int find_help_entry(char *form_menu_name, int field_menu_seq)
```

Synopsis:

This routine will search the active help structure (there needs to be only one). If the form_menu_name loaded does not match then help structure should be released and the appropriate one loaded. Normally, the using routine will call both the form and help load routines or menu and help routines at the same time. This would assure that whatever was active on the screen at the time had a supporting help construct. This routine will return -1 if no entry was found or the index of the table where the help starts.

display_help_entry

Prototype:

```
int display_help_entry(int index)
```

Synopsis:

This routine will build a window at fixed coordinates (we will get fancier next time) 5,5 with totalcolumns = MAX_LEN - 10, totalrows = MAX_LINES - 10, double border, top title = field_label or the value of the current menu line. There will be a bottom title (on the last row of the window inserted into the bottom border) which has an appropriate label for the terminal type (IBM type uses "PgDn, PgUp, ESC to return"). The idea is instruct the user which keys control scroll_down, scroll_up and ESC. The window is set up with scrollock(FALSE) because we want to control the scrolling manually. Each line of text stored in the help memory structure should then be written to the window using display_with_wrap. Note that any pre-existing NL characters or CR characters must be filtered out prior to passing the string to display_with_wrap. Loop until display_with_wrap returns an ERR (window full) or you reach end of help entry (the field_line_seq will have changed). Enter into a keyboard loop scrolling up or down as the user requests until the ESC key is entered. Once ESC is encountered erase the window and return to the calling program. Return is OK or ERR.

February 7, 1992

Curses Toolbox for ASV3

message_box

Prototype:

```
int message_box(char *msg, int nbr_button, char *button_text[])
```

Synopsis:

Refer to the general discussion under background for the Help Manager. The maximum number of lines possible to display with this routine will be `MAX_LINES - 19` (6 lines for 25 line terminal). The geometry constraints will be the same as indicated in the discussion on display help entry.

Internal Geometry of a message box:

_____title_____	line 1
	line 2
msg line 1	line 3
msg line 2	line 4
	line 5
	line 6
button	line 7
	line 8
	line 9
-----	line 10

line 1: border line/title

line 2: blank

line 3-4 msg[n] start message 2 columns right of the border, leave 2 cols on the other side

line 5: blank

line 6: button top border

line 7: button(s) text - try to balance around the center of the line

line 8: button bottom border

line 9: blank

line 10: bottom border line

User input consists of TAB, SHFT/TAB, and ENTER/RETURN. After the message box is built, the first button should be highlighted. Pressing ENTER/RETURN activates the button. TAB or SHFT/TAB returns the data attribute of the current button to normal and highlights the next or previous button.

Pressing ENTER/RETURN causes the window (message box) to be erased and all message box dependent memory to be released.

February 7, 1992

Curses Toolbox for ASV3

The routine will return the vales 1, 2, or 3 depending on which button was activated.

Menu Manager:

Background:

The primary job of the menu manager is to create a window with a scroll bar that the user can manipulate to select one or more options. Like the help manager, the window details are completely contained inside the menuing routines. The calling program need only reference a menu name and the menu routines will build the men and return the selections. Like the help menager, the text for each of the menu selections will be contained inside an ASCII file which must be accessed and loaded to memory. At this point we will not imbed any associated help logic inside the menu manager. The retains flexibility to either provide or not provide contextual help for menu options.

The internal memory structures supporting menu functions is as follows:

```
typedef struct
{
    WINDOW *menu_window;
    int ul_row;
    int ul_col;
    char *title;
    int type_border;
    int win_attr;
    int bord_attr;
    int menu_width;
    int menu_length;
    int menu_start;
    int menu_bar;
    TABLE *lines;
} MENU;
```

The following functions are contained in this module;

create_menu

Prototype:

```
int create_menu(char *src, int row, int col)
```

Synopsis:

February 7, 1992

Curses Toolbox for ASV3

The routine must create the menu construct and fill in all the missing pieces in the MENU structure. The first step would loading the menu source which can be done by calling `create_table_space` and passing it the pathway. The resulting TABLE pointer should be stored as the `lines` entry. The row and column parameters are used as the `ul_row` and `ul_col` entries respectively. The `menu_width` is `max_str_len + 4`. The `menu_length` is `MAX_LINES-row-4` or `nbr_lines` whichever is less. The type border is SINGLE, border attribute is normal. The title of the menu should be the menu name (in `src`). The bar attribute should be the inverse of whatever the menu attribute is (NORMAL for now). `menu_start` is the index of the menu line appearing at the top of the menu (0 for initial display) and the `menu_bar` is the index where the bar is currently sitting (0 for initial display). After initializing the MENU structure, `make_window` should be called. If all steps have been completed successfully return OK else return ERR.

`display_menu`

Prototype:

```
int display_menu(MENU *menu)
```

Synopsis:

Use the structure passed for determining what menu strings to display starting from the string index equal to `menu_start`. Use `display_with_chop` to display the menu strings within the window. Each string should be preceded by two spaces. This will support the multiple selection capability described in `get_selection`. Redisplay the string under the menu bar with the bar attribute (REVERSE for now). The top of the menu is at line 1 of the associated window (0 is the border). The last string to be displayed is at `menu_length-1` (`menu_length` is the bottom border). If you run out of strings before get to the `menu_length-1` line, display a line of spaces. This routine will return OK or ERR.

`get_selection`

Prototype:

```
int **get_selection(MENU *menu, char view)
```

Synopsis:

This routine controls the bar and scrolls the menu (for those menus longer than the window depth). Keys need to be defined for `scroll_down`, `scroll_up`, `arrow_down`, `arrow_up`, `top_of_menu`, `bottom_of_menu`, ENTER/RETURN, select (usually implemented using the space bar). Select will cause the first character of the `menu_bar` line (which had 2

February 7, 1992

Curses Toolbox for ASV3

leading spaces) to be changed from space to whatever the selected character is (a plus sign could work). Pressing the select key again should toggle the select character off. Pressing up_arrow when at the top_of_menu should "wrap" to the bottom of menu. Down_arrow at the bottom_of_menu should "wrap" to the top_of_menu. Pressing ENTER/RETURN should erase the menu and release all associated memory structures, then return an integer array of all the indexes selected. Pressing ESC should do the same thing but with a NULL integer array. As a special case in support of the directory manager there needs to be a View key defined. If the user pressed View and is supposed to be defined, then a call to display_file should be made. (See routines under Directory Manager)

Directory Manager:

The following functions are contained in this module;

get_directory

Prototype:

```
int get_directory(char *dir)
```

Synopsis:

This routine should indicated directory and make it the current directory. Return should be either OK or ERR.

get_files

Prototype:

```
int get_files(char *dir, char *files[], int mask)
```

Synopsis:

This routine should get all the files subordinate to the indicated directory and load their names and attributes (as defined by the mask variable) into the files array. Mask will be defined as a boolean combination of the following values; FILENAME, SIZE, PERMS, OWNER, DIR, LINK. The routine will return OK or ERR. Maximum for the files array will 500 for now.

display_file

Prototype:

```
int display_file(char *filename)
```

Synopsis:

February 7, 1992

Curses Toolbox for ASV3

This routine will attempt to access the file for read permissions after it has checked whether the file is ASCII. If the fopen fails or the file is not ASCII then the routine returns ERR. If the file is available use the create_table_space routine to access the file and load it into memory (use a max memory capacity of 500 lines). Create a window in a manner similar to display_help_entry (same constraints). Use display_with_chop to write the array to the screen. Support for scroll_down, scroll_up, and ESC should be provided. ESC will erase the window. The calling routine should reset focus and release the table array.

Screen Manager:

Background:

The overall flexibility of this system is dependent on the object-like nature of the various tool boxes. The primary function of the screen manager is to determine what types of display routines to be called based on the user's selected interface (Command-Line, Curses, or X). Input to the screen manager consists of the filename (character string) to the screen definition file. The screen definition file contains the default attributes of the screen, the form or menu definition file name, the form or menu item having initial focus or other high-level variables. The screen manager is called by the command manager and returns a command string to the command manager. The screen manager is also responsible for loading the appropriate help files (if present). In the case of the Curses and Command-Line options, the only difference is whether Forms/Menu Toolboxes or the Command_line Toolboxes are called. The inputs are identical as is the required memory structures and procedures. In other words, the Command-Line toolboxes uses the Curses toolbox structures but ignores screen geometry information. The X routines are completely different and stand alone from the rest of the system. They too will return the same types of parameters to the screen manager allowing the command manager to function identically regardless of the type of user interface in use.

Command Manager:

Background:

The command manager stands at the top of the system of toolboxes. The command manager will call the screen manager which will call the forms manager (or menu manager as appropriate) which calls the field manager. The field manager returns user data to the form manager which assembles it into a database record which it appends to the command string to be executed next and returns the whole assemblage to the command manager for disposition. If the menu manager were invoked (as opposed to the forms manager) then no field manager is required and

February 7, 1992

Curses Toolbox for ASV3

only a string is returned to the screen manager (usually a command for the command manager). There is an exception naturally. This is a form with a single CHOICE field. This is an intermediate construct similiar to the Command-Line interpretation of a menu. It looks like command line but allows information to posted below the input field. It is an emulation of the CEO "form". MountainNet's provided forms for ASV3.0 are very similiar to what they already have in CEO.

— Functional Decomposition of ASV3 requirements by Subsystem:

Library Management Subsystem (ASV3.0)	Email Subsystem:	Repository Services Subsystem:
User/Librarian Functions:	Read Mail	Problem Reports
—	Send Mail	Profile,
Browse	Maintain Mail List	Customization
Search	List ASV3 Users	Word Processing
View		Spell Checker
Check Out (Copy/Print to Home)		Calculator
		Password Changing
Librarian Functions:	File Transfer Subsystem:	System Maintenance
Catalog	Download	Source
Relate	Upload	Make
Object Classes	File Conversion	C Compiler
Assign Permissions		
Notify Users of Chgs		
Librarian Tools (ASV3.1??)		
—		
Evaluate		
Decompose		
Classify		
Librarian/User Tools:	System Admin Subsystem:	
—		
Ada Compiler	Archive	
Convention Checking (C, B, Lint)	Restore	
—	Establish Userid	
	Peripherals	

Functional Decomposition of ASV3 requirements by Type of User:

Customer Service Group:	Librarian Group:	User Group:
Browse	Catalog	Browse
Search	Relate	Search
View	Create/Delete Users	View
E-Mail		Download
Download		E-Mail
Conversion		Problems
		Profile/Defaults
System Administration Group:	Head Librarian Group:	Clean Room Group:
Archive	Object Class	Evaluate
Restore	Maintenance	Classify
UserID registration	Create/Delete	Decompose
	Librarians	Compile
Submitter Group:		
Conversion		
Convention Checking		

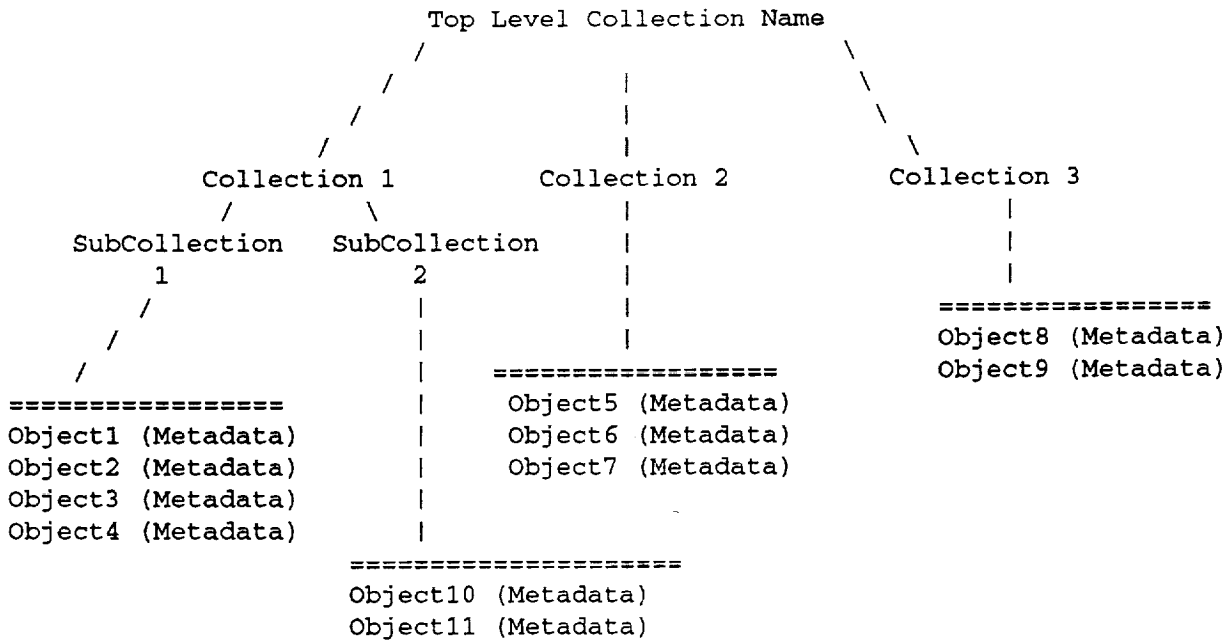
ORIGINAL PAGE IS
OF POOR QUALITY

Process Descriptions:

Cataloging:

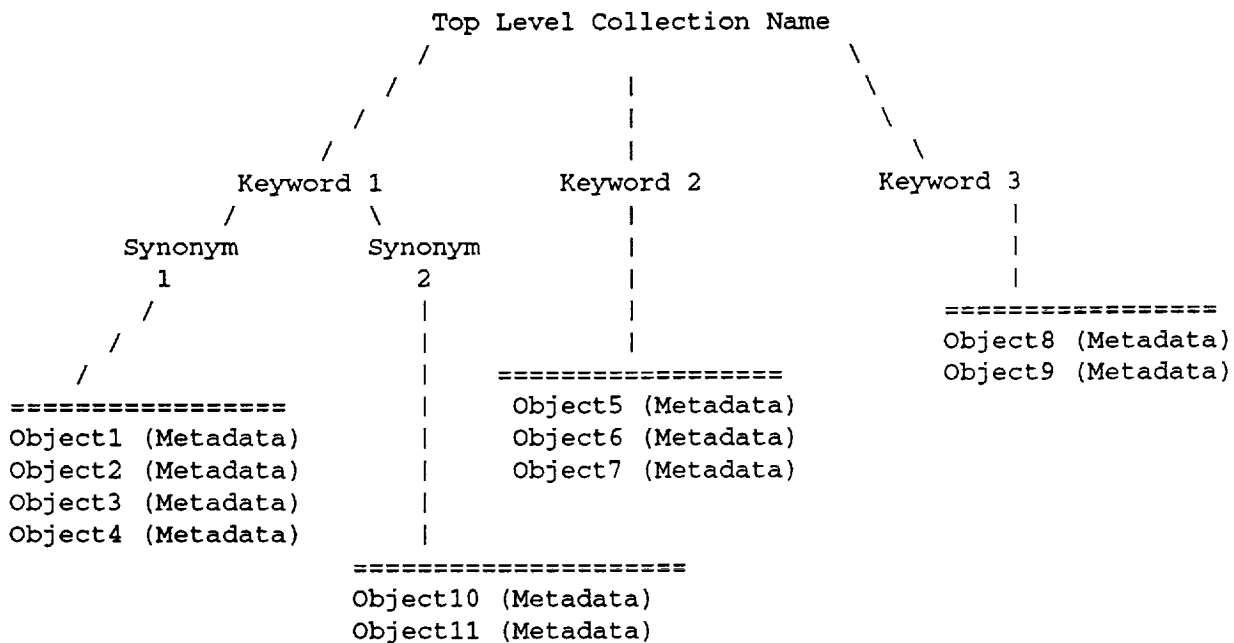
Cataloging is the process of assigning an object (separately stored file) to group name. This group name is known as a collection or subcollection within the autoLib 4 terminology. Collections may have more than one type of object (i.e. text files containing book reviews, GIF files showing the book cover and picture of the author) but may actually be restricted to one type. Type is also referred to "loosely" as an object class. The problem is that it is possible to have more than one type assigned to an object class. The precise definition of an object class is that metadata for all objects of the same class are stored in the same Oracle record format. The reason this metadata classification process is called object class is that object classes can be derived from other object classes. All objects have a global class, that is a minimal metadata set that stored in Oracle. New objects classes are data sets that are defined and stored in addition to the global set. Classes may be derived from other classes by requiring Oracle to store a record format for the global class, each of the "parents" of the derived classes and the new class. For example, say the global class required object name, submitter name, and object type. The conference class would add an additional record for the object consisting of conference date, conference location, and conference coordinator. A derived class from conference would have the global data elements, the conference data elements and any new data elements (ie. say we create a call-for-papers class which would add subject name, submission coordinator, and date-required-by). Oracle can store all this relationally, so that there are really three records which are defined within one set name. Where the relational model breaks down is that these three records are never allowed to exist independently from one another. Either all three exist or none exist. This is more closely modeled by the variable record length construct within COBOL where you define OCCURS ... DEPENDING ON... .

Model of Hierarchial Collections



As each of these collections are defined, representative keywords (aka Subjects) are established which are common to objects contained within specific collections (or subcollections). These keywords can have synonyms (aka Related Subjects). Optionally special keywords could be stored as part of each objects metadata. Collections have metadata too, the collection name, Subjects, Related Subjects, related collections, related subcollections, etc. Accessing by keyword (or Subject) finds the objects which have been related to that keyword. This could conceivable span several collections if keywords were assigned at the object level as opposed to the collection level. If keywords are assigned at the collection level, then there are a hierarchy of keywords which can be accessed.

Model of Hierarchial Keywords



Using the Combination of Collection Name and Subject/Related Subject provides a much higher probability that the infrequent user can successfully traverse the search paths to find what they are looking for. This technique illustrates multifaceted matrix classification. The librarian through hard-won practical knowledge of the library contents and the probable background of the user base artificially constructs the collection relationships, nomenclature, metadata associated (object classes) and metadata relationship (object hierarchical trees). This last capability can be illustrated by assuming the following class associations for the above objects:

Object	Class	Global Class
1	A	/
2	A	/ Class A =====+
3	B	/ Object1 (Metadata)
4	A	/ Object2 (Metadata)
5	C	Object4 (Metadata)
6	C	Object11 (Metadata) Class B
7	C	Object3 (Metadata)
8	B	Object8 (Metadata)
9	B	Object9 (Metadata)
10	C	Class C
11	A	Object5 (Metadata)
		Object6 (Metadata)
		Object7 (Metadata)
		Object10 (Metadata)

Walking the Object Class tree is like asking for list of phonograph records as opposed to a list of sheet music when you are looking over information related to music

It would be nice if one could change their mind after looking over the collection tree. Maybe subcollection 2 is more logically placed under collection 2 rather than collection 1. The head librarian is normally the maker of such policy level decisions and autoLib assumes that there is an equivalent level of user when such a reassignment is made. Reassigning the collections relationship affects the keyword hierarchy but not the object class hierarchy. Assuming that good abstracts of objects are included in the metadata, one need only provide metadata text search. The text search (on the metadata) is implemented as a natural language search or a boolean search. Within the original autoLib development environment so few objects were stored as text files that creating global text search (similar to GREP) was not considered. Additionally, global text searches rapidly consume computer resources. It wouldn't take many ongoing search processes to bring most networks to their knees.

The implications of all of the above is that the cataloging tools supports a diverse methodology of both classifying and describing objects. Full utilization of the tools for cataloguing would allow for successful search/access of objects most of the time, especially if the community has an element of uniformity about it.

Searching:

Searching is defined as using some type of keyword or natural language query. The keyword capability divides into boolean versus pattern matching. It should be emphasized at this point that this discussion describes searching the metadata stored about the object. Searching a text object can be accomplished using the GREP capability within UNIX. The metadata however is stored inside of Oracle and not available to UNIX utilities. All interface to Oracle is via the Structured Query Language (SQL) and is subject to the limitations of that language. In general SQL queries are composed of text or patterns of text which are or are not supposed to be resident in specified Oracle database fields. The software asks the user to fill in the blanks (fields, values, conditions) and creates the SQL query for the user. The results of the query are captured, formatted and displayed for the user. This allows more available power to the user without requiring the user to have mastered a query language. But Lib does not allow users knowing SQL to "do it themselves". This protects the elements (if there were any) which the Repository doesn't want the user to get at. This could be who had last checked an object, some system data useful to the librarians only or whatever. Natural Language queries are an attempt to let the user enter plain English questions and get what they were trying to access.

An important thing to remember about these searching techniques is that they are searching metadata that the librarians have entered. This methodology requires that the librarians analyse objects using the same techniques that would result in generation of cards for a card catalog within a normal library. Even the natural language query has this dependency.

Browsing:

Browsing is when the user using collection names as opposed to keywords to locate objects. It is far less efficient for the user but far more convenient for the librarian because much less work is required (no keywords, synonyms, etc.). The system allows for relationships to be identified and stored between collections, subcollections and objects. This is in addition to the implied relations of an object belonging to a particular subcollection/collection hierarchy. This relationship mechanism is called "linking" and is supported by use of a particular metadata field. Another method for browsing is searching alphabetically. One can list collections, subcollections, or objects alphabetically.

Viewing:

Viewing is simply displaying the contents of the object on the terminal. This is simple for text objects, more complex for others. This system comes with viewers for text, and three types of graphics formats. It is likely, that many others will be developed over time.

Relating:

Refer back to searching and the "linking" discussion.

Object Class Maintenance:

This process was alluded to in the cataloguing discussion. Basically, each type of object can have a unique set of metadata. That set is composed of a global or generic set common to all objects plus what ever is unique about a particular type. The system is supposed to support inheritance. This means that a new type (this is only an example) might differ from an existing type by a couple of metadata fields. The new type would be defined as inheriting the existing type (object class) and having the two unique metadata fields. This reduces the working in creating new types (assuming that good object oriented practice was used in defining existing types in the first place). One of the activities related to installation will be figuring out a good object oriented approach to types. It should be mentioned that autoLib restricts this type of activity to head librarians.

Evaluate:

This is a currently manual process where submissions are examined to determine whether they are ready for inclusion into the repository.

Classification:

This is the manual process of analysis of the submitted object which results in determination of what metadata is stored about the object and where the object fits into the collections scheme and how it relates to existing objects.

Decomposing:

This process (manual at this point) occurs when it is determined that the object is really composed of multiple pieces instead of a single entity. It would require followon evaluation and classification to relate the derived pieces with the existing library.

Conversion:

This process converts the object from one form to another. For example, it is common for modem users to convert text to a compressed form for uploading and downloading. It is also common for PC users to convert from one type of word processor to another and change graphics format from one type to another. All this requires software tools, as yet undefined.

Convention Checking:

Hopefully, the submitter can find some tools to run against his submission to flag things that the librarians would use to reject it. When we define what ever standards will be used to judge submissions, we can start defining these tools. A likely first step would be requiring the user to fill in metadata that would be appropriate (in the users opinion) to give the librarians a better place to start in the classification process.

Download/Upload:

This is simply transferring files from the repository host to or from somewhere else. Unfortunately, the UNIX community does it different from the desktop community. AutoLib will not support multiple file transfers so this function will reside in a subsystem exterior to autoLib but still inside the umbrella of ASV3.

Create/Delete Users:

There is only one class of user so no modify is required. This process defines a userid as a repository system user. It does not create userids on the UNIX system. The UNIX system administrator would create the user id. The librarian would grant library access permissions. The librarian will have some utilities to facilitate the creation of default scripts for the users that the UNIX system administrators do not have.

Create/Delete Librarian:

AutoLib allows many different levels of librarians. The only caveat is that a librarian can not grant library permissions in excess of what they themselves are allocated. The head librarian has all the permissions. There can be more than one head librarian.

Archive/Restore:

This is the process of saving objects offline or putting the objects back on line. This is a two phase process (at this point) where the system administrator controls what is online or offline and the librarian has to update the metadata to indicate where the object is. Cooperation is essential.

Profile/Defaults:

The user can tailor some things about the autoLib behavior as well as other ASV3 utilities. Typical UNIX conventions will be honored. This is not to imply that ASV3 users will have access to UNIX. The details of this are in development.

Problem Reporting:

This is a subsystem where users can report problems in a way that allows tracking and direct feedback to the user (if that seems desirable). It will have a form to file out. It is external to autoLib. It will use Oracle to store the problem reports.

E-Mail:

At this point, ASV3 is presumed to be using Oracle Mail. It supports all of the current ASV2 functionality using less disk space for mail lists. This may change if there are schedule/delivery problems from the vendor.

What is left out?

There are no defined reports at this point. Oracle is pretty good with reports. Usage of the system is tracked in Oracle for autoLib functions. The rest of the ASV3 system could use the same mechanisms. However, there is a lot of reports besides usage statistics. These need to be defined.

The process of Cataloguing could be broken down as follows:

Process Name Subsystem	ASV3 Requirement	AutoLib 4.0 Requirement	ASV3
Allocated To:	Allocated To:	Allocated To:	
Add Objects 4.0	5.1.1.1.2.b	4.3.5.a	autoLib
* Modify/Update Objects CM/Version Ctrl	5.1.1.1.2.b	shortfall	
	5.1.2.2.c		
	5.1.2.2.d		
Delete Objects 4.0	5.1.1.1.2.b	4.3.5.c	autoLib
Link Objects 4.0	5.1.1.4.k	implied only	autoLib
(via mewtadata)	5.1.1.4.1		
Add Tools (acting on 4.0	5.1.2.2.e	4.3.4.a	autoLib
objects)	5.1.2.2.f		
* Modify/Update Tools CM/Version Ctrl	5.1.2.2.e	shortfall	
	5.1.2.2.f		
Delete Tools 4.0	5.1.2.2.e	4.3.4.c	autoLib
	5.1.2.2.f		
Add Object Metadata 4.0	5.1.1.1.2.e	4.3.5.d	autoLib
	5.1.1.3.b		
	5.1.1.3.c		
	5.1.1.3.1		
Modify/Update Object 4.0	5.1.1.1.2.e	4.3.5.e	autoLib
Metadata			
* Delete Object Metadata CM/Version Ctrl	5.1.1.1.2.e	shortfall	
Add Collections	5.1.1.1.2.c	4.3.3.a	
	5.1.1.1.2.d		
* Modify Collections 4.0	5.1.1.1.2.c	shortfall	autoLib
	5.1.1.1.2.d	more than just moving	
Delete Collections 4.0	5.1.1.1.2.c	4.3.3.c	autoLib
	5.1.1.1.2.d		
Link Collections 4.0	5.1.1.4.k	4.3.3.e	autoLib
	5.1.1.4.1		
Add Keywords (Subjects)	5.1.2.2.k	4.3.6.a	

* Add Related Subjects	5.1.1.1.2.c	uncertain
	5.1.1.1.2.d	maybe part of NLQ
Modify Subjects	5.1.1.1.2.c	4.3.6.c
	5.1.1.1.2.d	
Delete Subjects	5.1.1.1.2.c	4.3.6.b
	5.1.1.1.2.d	
* Link Subjects	5.1.1.4.k	uncertain
	5.1.1.4.l	
* Modify Related Subjects	5.1.1.1.2.c	uncertain
	5.1.1.1.2.d	
* Link Related Subjects	5.1.1.4.k	uncertain
	5.1.1.4.l	
* Delete Related Subjects	5.1.1.1.2.c	uncertain
	5.1.1.1.2.d	

The process of Searching/Accessing could be broken down as follows:

Process Name Subsystem	ASV3 Requirement	AutoLib 4.0 Requirement	ASV3
Allocated To:	Allocated To:	Allocated To:	
Search For Objects using AutoLib 4.0 Metadata	5.1.1.1.1.a	4.2.2.a - d	
	5.1.1.1.1.b	4.2.2.1.a - d	
	5.1.1.1.1.c	4.2.2.2.a - g	
	5.1.1.1.1.d	4.2.2.4.a - d	
	5.1.1.1.1.e		
Search For Objects using Browser Text	5.1.1.1.1.k	shortfall	Batch
Search For Objects using 4.0 Collection/Subject	5.1.1.1.1.g	4.2.1.1.a - m	autoLib
	5.1.1.1.1.h		
	5.1.1.1.1.i		
* Search Again using last or edited criteria	5.1.1.1.1.f	unclear probable shortfall	
View Objects 4.0	5.1.1.1.1.j	4.2.1.2.a - r	autoLib
	5.1.1.1.1.l		
Copy Objects to workarea 4.0	5.1.1.1.1.m	4.2.1.2.f	autoLib
	5.1.1.1.1.o		
	5.1.1.1.1.p		
	5.1.1.4.g		
	5.1.2.1.1		

* Dump Object to Specified Media

shortfall in spec

	5.1.1.4.g		
Getting Help	4.3.a	4.1.a	all
subsystems	4.3.b	4.1.b - j	
	4.3.c		
	4.3.c.1		
	4.3.c.2		
	4.3.d		
	4.3.e		
	4.3.f		
	4.3.g		
	4.3.h		
	4.3.i		
	4.3.j		
	4.3.k		
	4.3.l		
	4.3.m		

Other User Processes required by ASV3:

Process Name Subsystem	ASV3 Requirement	AutoLib 4.0 Requirement	ASV3
Allocated To:	Allocated To:		
Signing on to ASV3 level	4.1.a	4.1.a	system
	4.1.b		
	4.2.a		
	5.1.1.2.1.f		
	5.1.2.2.a		
	5.1.2.2.m		
Download Files via modem	4.1.b	4.5.i	
	4.1.c	Kermit Only	
	4.2.d		
	4.2.b		
	5.1.1.1.1.n		
Download Files via FTP	4.1.a	4.5.j	
	4.2.a		
	5.1.1.1.1.n		
Submission	5.1.1.1.2.a	n/a	Misc
Qualify Submission (3.1)	5.1.1.1.2.e	n/a	Misc
Using Electronic Mail	5.1.1.1.1.q	n/a	E-Mail
	5.1.1.1.1.r		
	5.1.1.1.1.s		
	5.1.2.2.n		

Reporting Problems	5.1.1.4.e	n/a
Repository Services	5.1.1.4.i	ideally should capable within autoLib too

System/Network Managers Processes required by ASV3:

Process Name Subsystem	ASV3 Requirement	AutoLib 4.0 Requirement	ASV3
Allocated To:	Allocated To:		
Security Procedures all subsystems	5.1.1.2.c	4.3.1.i	affects
	5.1.1.2.d	4.4.a - d	
	5.1.1.2.e		
	5.1.1.2.f		
	5.1.1.2.g		
	5.1.1.2.h		
Establish User Resources 4.0	5.1.1.2.a	4.3.1.h	autoLib
	5.1.1.2.b	this may not be the right	
place			
Establish Librarian Resources	5.1.1.2.a	4.3.1.c - f	
place		this may not be the right	
	5.1.1.2.b		
	5.1.1.2.h		
Establish Head Librarian ASV3 req		4.3.1.a, b, g	not
Resources		4.3.2.a	
Archival Offline Object	5.1.1.4.m	n/a	
Repository Services (Audit Trail)	5.1.1.4.n		
Archival Offline Subject	5.1.1.4.m	n/a	
Repository Services (Audit Trail)	5.1.1.4.n		
Archival Offline Collection	5.1.1.4.m	n/a	
Repository Services (Audit Trail)	5.1.1.4.n		
Backup Online Object	5.1.1.4.j	n/a	
Repository Services (Audit Trail)	5.1.2.1.m		
	5.1.2.2.o		
Backup Online Subject	5.1.1.4.j	n/a	
Repository Services (Audit Trail)	5.1.2.1.m		
	5.1.2.2.o		

Backup Online Collection	5.1.1.4.j	n/a	
Repository Services (Audit Trail)	5.1.2.1.m		
	5.1.2.2.o		
Restore Online Object	5.1.1.4.j	n/a	
Repository Services (Audit Trail)	5.1.2.2.o		
Restore Online Subject	5.1.1.4.j		
Repository Services (Audit Trail)	5.1.2.2.o		
Restore Online Collection	5.1.1.4.j	n/a	
Repository Services (Audit Trail)	5.1.2.2.o		
Monitor System Utilization	5.1.1.1.2.f	4.4.a - e	TBD
	5.1.1.1.2.g		
Migrate ASV2 data into ASV3 standalone batch	5.1.1.3.a	n/a	

APPENDIX F
DELIVERABLES COVER SHEET

RB.05 GHG Corporation
January 1, 1992

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW # 3

Title of Task: Draft Help Screen Formats - memo to K. Fleming

Subcontractor: GHG Corporation

Cooperative Agreement No. NCC-9-16

Principal Investigator: E.T. DICKERSON

NASA Technical Monitor: E. FRIDGE

Type of Report: _____

Period Covered by Report _____

Due Date: 03/17/92

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

17 March 1992

To: Karen Fleming
From: Dave Henning

Re: Draft Help Screens for Output Request Function

Enclosed are the initial drafts of the Output Request Help Screens. These will display to the user when HELP is pressed in various windows. I am more concerned about tone here. These helps are stored as individual files in clear text and can be edited or changed at any time. I suggest that tailoring for POC on copy and special instructions for printing be put off until we can actually install all of this at your site. The power of this system is its vulnerability also. The power is in its invocation of script and configuration files. That means a lot of initial set up work for the System Administrator and the Oracle Database Manager. I would like to reiterate that you should be seriously thinking of who these individuals are going to be and start make arrangements for any special training that they might need in UNIX and Oracle. Naturally, the RS-6000 is not a standard UNIX system so additional IBM specific training will be needed if that is the hardware selection.

Let me know if you have serious problems with the screens.

Dave Henning
GHG Corp

Request Output

The Request Output function is provided so that the user may "check out" items from the software repository. The functions provided are:

- | | |
|------------|---|
| Copy | Request a copy of an object or objects be provided on a specific media (see Request Output - Copy for more detail) |
| Download | Download the object and its associated metadata to your system via one of the supported protocols (see Request Output - Download for more detail) |
| Print | Queue the print of an object and its associated metadata to one of the repository printers. Objects printed out will be forwarded to the requester at the address listed in the Client data file. |
| Local Copy | Provided as a Librarian Function only. This will make a physical copy of requested objects and their associated metadata in the requesters HOME directory. |

The system will record the check out request for each object selected. The repository will notify the requesters (as resources are available) when new versions of object are received or when problem with objects are uncovered. These functions support session requests. This means that after you select the objects and provide whatever data is necessary to complete the output request, you will be asked whether this should be done immediately or later. If you choose later, then object requests will accumulated throughout your session. You may mix print, copy, and download requests and still take advantage of this session capability. The only cost to you is an extended signoff. All stored output request actions are executed prior to logoff.

Request Output - Now or Later

This message box is asking you whether you want to generate your media request or download request now or later. The system will keep track of all the objects you have marked during your session and transmit them at the end of the session at signoff if you would rather do that. Select LATER to postpone the output request execution but to keep track of what was selected. Select NOW and the system will prompt you for appropriate information to execute the output request. The output request is then done in the background while you can continue working.

Request Output - Copy

This function is provided so that users may check out objects and receive them on a specified media. The particular media supported at the moment includes:

- 1 3.5 High Density IBM or PC
- 2 3.5 Low Density IBM or PC
- 3 3.5 MAC
- 4 5.25 High Density IBM or PC
- 5 5.25 Low Density IBM or PC
- 6 9 Track 1600 BPI tape (TAR)
- 7 9 Track 1600 BPI tape (ASCII)
- 8 1/4 inch Cartridge Tape
- 9 Hardcopy (14.5 by 11)
- 10 Hardcopy (8.5 by 11)

Multiple objects may be requested at a time. The repository staff will copy the objects on the requested media and ship the results to the requester using mailing information in the Client data file. This is usually the most convenient and fastest way to check objects out of the repository.

Request Output - Copy - 3.5 High Density IBM or PC

This function requests repository staff to physically copy your selected objects and the associated metadata to a 3 1/2 inch high density (1.44M) diskette readable on an IBM type computer.

Request Output - Copy - 3.5 IBM or PC

This function requests repository staff to physically copy your selected objects and the associated metadata to a 3 1/2 inch low density diskette (720K) readable on an IBM type computer.

Request Output - Copy - 3.5 MAC

This function requests repository staff to physically copy your selected objects and the associated metadata to a 3 1/2 inch diskette readable on an Macintosh type computer.

Request Output - Copy - 5.25 High Density IBM or PC

This function requests repository staff to physically copy your selected objects and the associated metadata to a 5 1/4 inch high density (1.2M) diskette readable on an IBM type computer.

Request Output - Copy - 5.25 IBM or PC

This function requests repository staff to physically copy your selected objects and the associated metadata to a 5 1/4 inch low density diskette (360K) readable on an IBM type computer.

Request Output - Copy - 9 Track 1600 BPI tape (TAR)

This function requests repository staff to physically copy your selected objects and the associated metadata to a 9 track 1600 BPI Reel of computer tape in a TAR format.

Request Output - Copy - 9 Track 1600 BPI tape (ASCII)

This function requests repository staff to physically copy your selected objects and the associated metadata to a 9 track 1600 BPI Reel of computer tape in a ASCII format.

Request Output - Copy - 1/4 inch Cartridge Tape

This function requests repository staff to physically copy your selected objects and the associated metadata to a 1/4 inch Cartridge Tape in a TAR format.

Request Output - Copy - Hardcopy (14.5 by 11)

This function will print objects or at least attempt to print objects on the a HOST printer using a 14.5 x 11 paper size. If the object format is incompatible with the device requested, then the system will return an error. The protocol used to check compatibility is not 100% reliable so the user needs to exercise some care with this function.

Request Output - Copy - Hardcopy (8.5 by 11)

This function will print objects or at least attempt to print objects on the a HOST printer using a 8.5 x 11 paper size (landscape mode). If the object format is incompatible with the device requested, then the system will return an error. The protocol used to check compatibility is not 100% reliable so the user needs to exercise some care with this function.

Request Output - Download

This function is provided to enable users to download selected objects via FTP, Kermit, or Zmodem. In addition objects can be mailed via UUCP. The specific option available within this function are as follows:

- 1 FTP
- 2 KERMIT
- 3 X-MODEM
- 4 Y-MODEM
- 5 Z-MODEM
- 6 UUCP

The system will record the check out request for each object selected. The repository will notify the requesters (as resources are available) when new versions of object are received or when problem with objects are uncovered. See Request Output - Download - option for more specific detail on each option.

Request Output - Download - FTP

This function will prompt for the target site address and your userid/password to access that site. This function transfers files to the target site. You must have a valid userid/password at the site and know the correct internet address of the site.

You will be prompted for your USERID. This is the userid you use to sign the target system.

The PASSWORD is the password that you use to sign on the target system.

The TARGET is the internet address in either alpha (asv3.wvnet.edu) form or numeric (129.71.42.1) form.

After you have entered all the data, press the OK button and the system will attempt to sign on and transfer all the objects you have selected. All objects will be transferred to the home directory of your userid on the target system.

Request Output - Download - UUCP

This function will prompt for the target site address (a so-called bang address). This function transfers files in the background to the target site. This similiar to E-mail. All files will end up in the UUCPPUBLIC directory at the target site. The site administrator can

—
tell you precisely where that is. This directory has general
permissions so you can read and retrieve the objects you transmit.
—

—
The TARGET is the internet address of the form
user@target!target!target
—

After you have entered all the data, press the OK button and the
system will attempt to sign on and transfer all the objects you have
selected.
—

Request Output - Print

This function will print objects or at least attempt to print objects on the requested HOST printer device. If the object format is incompatible with the device requested, then the system will return an error. The protocol used to check compatibility is not 100% reliable so the user needs to exercise some care with this function.

The printer devices currently supported are:

- psghg - a Post Script printer located in the GHG Lab
- lpghg - a Line Printer located in the GHG facility

Remember that the objects come out on the repository printer so that the repository staff can mail the results to you. They will use whatever mailing information is resident in the Client data file. If you want to check what this file says exit to the shell and access the e-mail function.

Request Output - Local Copy

This function is provided for the convenience of the repository staff. Objects selected will be physically copied to the requester's HOME directory. Non-staff users can not access their HOME directory directly so they will be excluded from using this function.

What would you like to do with the Session File?

This question indicates that you have requested output but delayed its creation. You may select CANCEL which then throws away your output requests or you may select PROCESS which will implement your output requests.

APPENDIX F
DELIVERABLES COVER SHEET

RB.05 GHG Corporation
January 1, 1992

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW # 4

Title of Task: History Table request - memo to M. Revig

Subcontractor: GHG Corporation

Cooperative Agreement No. NCC-9-16

Principal Investigator: E.T. DICKERSON

NASA Technical Monitor: E. FRIDGE

Type of Report: _____

Period Covered by Report _____

Due Date: 03/18/92

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

18 Mar 1992

To: Mark Rorvig, STB NASA

Cc: K. Fleming, MountainNet
J. McGee, GHG RBSE Project Manager
C. McKay, RBSE Chief Scientist
E.T. Dickerson, RBSE Program Manager

From: David L. Henning, GHG Corporation

Subject: The History Table

Background:

Various actions can be logged as a consequence of operating NELS. These are recorded via an instrumentation hook known as record_action. At this point in the development cycle for NELS, the following actions are being recorded.

Action	Associated Data
Log On	store userid, datetime stamp
Log Off	store userid, datetime stamp
Notify All	store userid, datetime stamp
Create Collection:	store collection_id and datetime_stamp
Modify Collection:	store collection_id and datetime_stamp
Remove Collection:	store collection_id and datetime_stamp
Create Object:	store userid, collection_id, object_id, datetime_stamp
View Object:	store userid, collection_id, object_id, datetime_stamp
Duplicate Object:	store userid, collection_id, object_id, datetime_stamp
Print Object:	store userid, collection_id, object_id, datetime_stamp
Modify Object:	store userid, collection_id, object_id, datetime_stamp
Delete Object:	store userid, collection_id, object_id, datetime_stamp

Problem:

The reporting requirements of MountainNet imply that the following additional actions need to be recorded. All of the following actions require recording userid, objectid, collectionid, datetime stamp. The download and media request activities require saving the requested media type or type of download protocol requested.

Request Media Object	
Request Download Object	
Archive Object	
Archive Collection	
Request Media File	Actions associated with objects having
Request Download File	multiple files associated with them

View File	
Print File	

A specific history table format was requested by MountainNet in a memo faxed to Mark Rorvig, STB, NASA. The format also included the following actions (in addition to what is identified above):

Action	Requested Action Detail

Browser Entry	how the user got here (search, menu, what) collection id matrix of object ids displayed in the browser datetime stamp userid
Browser Exit	datetime stamp userid
Metadata Display Entry	object id datetime stamp userid
Metadata Display Exit	object id datetime stamp userid
Abstract Display Entry	object id datetime stamp userid
Abstract Display Exit	object id datetime stamp userid
Viewer Entry	object id datetime stamp userid
Viewer Exit	object id datetime stamp userid
Directory Viewer Entry	object id datetime stamp userid
Directory Viewer Exit	object id datetime stamp userid
File Viewer Entry	object id file pathway type of file userid datetime stamp
File Viewer Exit	object id file pathway type of file userid

Search Entry	datetime stamp type of search userid datetime stamp
Search Exit	type of search userid datetime stamp
Search Activity	type of search userid datetime stamp
Log On	query string entered by the user userid datetime stamp
Log Off	mode (X, ASCII, what) userid datetime stamp cpu, i/o, etc. statistics

This last group of instrumentation was specifically requested by Dr. Eichmann to support his research efforts on repositories. These actions are also the reason for the request for a new history table format since they require capture of state data in addition to what is normally captured from the library manager.

The supporting structure requested was:

```
userid
Unix datetime stamp
object id
collection id
action code
action detail - 240 character field (free form apparently)
```

The current action logging is insufficient for MountainNet reporting requirements but could be accommodated in the existing History Table without the requested structure change. Dr. Eichmann's requests are more far reaching and definitely require a structure change. GHG will assume responsibility for adding the action recording interface (record_action) to the request output code. NASA needs to add the recording interface for the following actions to satisfy MountainNet recording requirements.

Action	Associated Data
Archive Object:	store userid, collection_id, object_id, datetime_stamp
View File:	store userid, collection_id, object_id, datetime_stamp
Archive Collection:	store collection_id, datetime_stamp

This requires that additional action codes be identified to support the following actions:

Archive Object
Archive Collection
Request Media Object
Request Download Object
View File
Request Media File
Request Download File
Request Print File

If the multiple file objects are accessed via anything other than the standard object browser then GHG must be made aware of how this works so that the GHG Extensions can be added to the special browser as well.

GHG will assume responsibility for adding the following action recording after the supporting codes have been assigned (see above):

Action	Associated Data
Request Media Object	userid, collection_id, object_id, datetime_stamp, media
Request Download Object	userid, collection_id, object_id, datetime_stamp, proto
Print Object:	userid, collection_id, object_id, datetime_stamp, media

When GHG is required to add extensions to the special object browser then GHG will also add logging for the following:

Action	Associated Data
Request Media File	userid, collection_id, object_id, datetime_stamp, media
Request Download File	userid, collection_id, object_id, datetime_stamp, proto
Print File:	userid, collection_id, object_id, datetime_stamp, media

If these requirements can not be supported then this must be identified. If these requirements affect schedule then this must be identified. GHG is aware that modifications to structure are considered out of scope for NELS 1.2, however some thought should be given to how NELS could support Dr. Eichmann's requirements and when that might happen.

CAVEAT: Note that the current system does not seem to differentiate between a Copy Object Metadata (copying object to another collection) and Copy Object to Directory (Physical copy of object with no metadata). Also note that the current action PRINT_OBJECT references calls to the tool interface rather than the GHG extensions. If there are intended to be more than one way to print or copy an object then there should be some differentiation in the recording mechanism.

The following potential duplication in recording exists:

GHG Local Copy and a NELS Copy (via tool)
GHG Print and a NELS Print (via tool)
NELS Copy (via tool) and a NELS Copy Object to another Collection

There may be others also.

EXHIBIT E
DELIVERABLES COVER SHEET

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW #5

Title of Task: ASV3 Requirements Document

Subcontractor: GHG

Cooperative Agreement No. NCC 9-16

Principal Investigator: E. T. Dickerson

NASA Technical Monitor: E. Fridge

Type of Report: Report/Software

Period Covered by Report 1/92 - 3/92

Due Date: 5/31/92

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

**ASV3 Requirements Document
of the
Repository Based Software Engineering (RBSE) Program**

ADANET-FD-R&T-093-0

MARCH 11, 1992

**Subcontract No. 044
Cooperative Agreement NCC9-16
Project No. RICIS No. SE.18**

**Submitted to:
University of Houston - Clear Lake
2700 Bay Area Boulevard
Houston, TX 77058-1096**

**Prepared by:
GHG Corporation
1300 Hercules, Suite 111
Houston, TX 77058**

**MountainNet, Inc.
P.O. Box 370
Dellslow, WV 26531-0370**

**ASV3 Requirements Document of the
Repository Based Software Engineering (RBSE) Program**

ADANET-FD-R&T-093-0

March 11, 1992

APPROVED BY :

Ernest M. Fridge
National Aeronautics and Space Administration (NASA)
Information Systems Division (ISD)

E. T. Dickersen
Program Manager
Research Institute for Computing
and Information Systems (RICIS)
University of Houston-Clear Lake

J.O. McGee
Program Manager
GHG Corporation

TABLE OF CONTENTS

Section	Page
1.0 INTRODUCTION	1-1
1.1 Identification and Scope.....	1-1
1.2 Purpose and Objectives	1-1
1.3 Status	1-1
1.4 Organization.....	1-1
2.0 RELATED DOCUMENTATION.....	2-1
2.1 Parent Document	2-1
2.2 Applicable Documents	2-1
3.0 REQUIREMENTS APPROACH AND TRADEOFFS.....	3-1
4.0 EXTERNAL INTERFACE REQUIREMENTS.....	4-1
4.1 Physical Interface Requirements	4-1
4.2 Software Interface Requirements	4-1
4.3 Human Interface Requirements	4-1
5.0 REQUIREMENTS SPECIFICATION.....	5-1
5.1 Process and Data Requirements	5-1
5.1.1 Library System Requirements	5-1
5.1.1.1 Functionality Requirements.....	5-1
5.1.1.1.1 Usage Functions.....	5-1
5.1.1.1.2 Maintenance Functions.....	5-2
5.1.1.2 Security and Privacy Requirements	5-3
5.1.1.3 AdaNET Holdings.....	5-4
5.1.1.4 Operational Requirements	5-5
5.1.2 Host Computing System Requirements.....	5-7
5.1.2.1 Hardware Requirements.....	5-7
5.1.2.2 Commercial Software Requirements	5-8
5.2 Performance and Quality Engineering Requirements	5-9
5.2.1 Capacity and Performance Requirements	5-9
5.2.2 Environmental Requirements	5-10
5.2.3 Quality Assurance Requirements.....	5-10
5.3 Safety Requirements	5-11
5.4 Implementation Constraints	5-11
5.5 Site Adaptation	5-12
6.0 TRACEABILITY.....	TBD
7.0 PARTITIONING FOR PHASED DELIVERY.....	7-1
8.0 ABBREVIATIONS AND ACRONYMS	8-1
9.0 GLOSSARY	9-1
10.0 APPENDIX A - CURRENT FUNCTIONALITY OF ASV2.....	10-1

1.0 INTRODUCTION

1.1 Identification and Scope

This document specifies the requirements for version 3 of the AdaNET system which is known as AdaNET Service Version 3 (ASV3). The structure and content of the document are consistent with the Software Management and Assurance Program (SMAP) Information System Life-Cycle and Documentation Standards, Release 4.3, and have been tailored from SMAP-DID-P200-SY.

1.2 Purpose and Objectives

This document defines the detailed requirements necessary to guide the production of ASV3. It therefore serves as the basis for four activities: preparation of a Request for Proposal (RFP) for new hardware, preparation of Requests for Quotation (RFQ) for appropriate software, adaptation and installation of autoLib, and operations planning for ASV3.

ASV3 is an interim replacement for ASV2 to be based on the autoLib library management system (see the AdaNET Program Management Plan, section 3.1.1). As such it is to maintain the preferred functionality of ASV2 providing significantly improved performance and a more usable interface. All design and implementation must be traceable to this document. Features which cannot be reasonably derived from these requirements will either be out of scope or will necessitate a change to the specifications contained in this document.

1.3 Status

After formal review and approval this document will be baselined and placed under configuration control. Any changes to the ASV3 requirements must then follow program configuration management change procedures, including formal review and rebaselining.

1.4 Organization

No sections of this document are rolled out; the document is self contained. The requirements for the ASV3 hardware and software platform are based in part upon the requirement to use NELs 1.2 and the requirements that are in Sections 4 and 5, EXTERNAL INTERFACE REQUIREMENTS and REQUIREMENTS SPECIFICATION. Section 6 presents traceability of each requirement to other material dictating the form and content of the ASV3 system.

2.0 RELATED DOCUMENTATION

2.1 Parent Document

ASV3 is a replacement for ASV2 with no specific extensions in functionality. Consequently this document is derived from the AdaNET Program Management Plan (PMP), March 21, 1992, Subcontract No. 044, Cooperative Agreement NCC9-16, Project No. RICIS No. SE.18.

2.2 Applicable Documents

- a. Design and Implementation Document, AdaNET Service Version 2.0 (ASV2), April 4, 1989.
- b. AdaNET Service Version 2.0 User's Guide, November 19, 1990.
- c. AdaNET Operations Team Monthly Report, ADANET-FD-O&D-044-1.
- d. Review of Current ASV2 Software and Documentation, Final Draft, September 17, 1990, Subcontract No. 044.
- e. Analysis of ASV2 Utilization, May 3, 1991, AdaNET-FD-R&T-087-0.
- f. NELS Version 4.0, Software Requirements Specification, JSC-24760, Information Technology Division, National Aeronautics and Space Administration, Lyndon B. Johnson Space Center, November 1990.
- g. Approved Change Requests to NELS 4.0, as detailed in appendix C.
- h. NELS ANSI User Interface Evaluation, April 12, 1991, Lionel Hanley, GHG memorandum to Bob Hennan of Barrios Technologies, ADANET-FD-R&T-080-0.
- i. NELS Requirements and Concerns, April 19, 1991, Karen J. Fleming, MountainNet memorandum to Lionel Hanley of GHG Corporation.
- j. NASA Thesaurus, 1988 edition, NASA SP-7064.
- k. IEEE Glossary of Software Engineering Terminology, 729-1983.

- l. Version 1 of U.S. Government Open Systems Interconnection Profile (GOSIP); Federal Information Processing Standard (FIPS) 146, August 15, 1990.
- m. Requests for Quotation on a medium-sized, Unix based, client server open architecture, GOSIP Compliant computer system to support the RBSE Program and the AdaNet Repository, December 17, 1991.
- n. The Program Management Plan of the Repository Based Software Engineering (RBSE) Program, RBSE-FD-PO-003-1; March 21, 1991.

3.0 REQUIREMENTS APPROACH AND TRADEOFFS

ASV3 must provide a more responsive and easier to use system for the period during which ASV4 is being specified, designed, and implemented. Aside from these useability objectives, there are no requirements for extending the capabilities of the ASV2 system; however, enhanced functions may be derived naturally through the process of replacing the hardware and software platforms which host the system; and compliance with the GOSIP FIPS 146. In addition, the ASV3 repository intends to evolve, with the delivery of upgraded versions, in compliance with the government schedules for evolving GOSIP standards.

The requirements contained in this document were derived from the functionality of ASV2, user feedback, and by focusing on issues and deficiencies identified in the evaluation and platform studies presented in document references c and d in section 2.2. Thus a combination of observations and wishes expressed by users, as documented in the evaluation, and guidance from earlier studies drove the derivation of the requirements.

Given the directives from the PMP and the capabilities of the designated NELS system, many issues were reduced in scope. For example, cost, risk, and compatibility concerns limited hardware and system platform options and stipulated some commercial-off-the-shelf (COTS) supporting products like Open Software Foundation (OSF) Motif.

The major influence on these requirements was the decision to host ASV3 with NELS. Given this direction, it was not necessary to provide a comprehensive set of requirements for a library system. Those functions native to NELS are mandated by definition and are not repeated here except where beneficial to subsequent acquisitions and efforts. New requirements necessary to accommodate ASV3 and to satisfy program objectives are presented in detail.

4.0 EXTERNAL INTERFACE REQUIREMENTS

4.1 Physical Interface Requirements

- a. ASV3 shall be accessible to end users and submitters nationwide via public networks, including, but not limited to, Internet, SprintNet, and WVNET.
- b. ASV3 shall be accessible via direct dial up modems.
- c. AdaNET direct dial-up capabilities shall support 300, 1200, and 2400 baud transmission rates using standard ASCII characters and ANSI X3.40, X3.41, and X3.64 modes.
- d. The following modem protocol standards shall continue to be supported:

300 baud	Bell 103
1200 baud	Bell 212A and CCITT V.22
2400 baud	CCITT V.22 bis and MNP 5

4.2 Software Interface Requirements

- a. ASV3 shall allow access through both X-window and ANSI terminal emulation standards.
- b. ASV3 shall provide file transfer via Kermit, File Transfer Protocol (FTP), XMODEM, YMODEM, ZMODEM, and UUCP.

4.3 Human Interface Requirements

- a. DELETED
- b. A standard, main menu shall always be presented to the user after log on.
- c. ASV3 shall provide comprehensive, universally accessible, context-sensitive, online help. This shall include providing help immediately from current program options and also the ability to obtain practical assistance from anywhere in the system.
- d. Instructions and prompts will be consistent throughout the system.

- e. Instructions and prompts will remain on the screen until acknowledged by the user.
- f. Displays shall be consistent as to format and to the naming, placement, and assignment of keys.
- g. The ANSI terminal interface shall implement all functions using the ANSI X3.154-88 standard. Additional keys may be implemented for convenience, for example, editing, cursor movement, and function keys, provided they conform to the conventions of the standard IBM PC 84-key keyboard.
- h. A consistent set of "short cut" keys shall be universally available. For example users should be able to use a single key to abandon an activity, to obtain help, or to return to the main menu from anywhere in the menu hierarchy.
- i. A status line shall indicate which standard and navigational keys are applicable to any location within the menu hierarchy.
- j. Functions of the ANSI terminal emulation will be implemented in such a way as to minimize the number of keystrokes.
- k. Data entry shall be directly into the corresponding field of those displays requiring user input values.
- l. All windows, pull-down menus, and messages which overlay parts of a display shall be clearly delineated from the underlying screen contents.
- m. Operations generally requiring response time greater than 3 seconds should present a status indication to the user.

5.0 REQUIREMENTS SPECIFICATION

5.1 Process and Data Requirements

5.1.1 Library System Requirements

5.1.1.1 Functionality Requirements

5.1.1.1.1 Usage Functions

- a. ASV3 shall support searching on object attributes as indicated in section 5.1.1.3.
- b. ASV3 shall support the viewing of lists of attributes such as keywords.
- c. ASV3 shall support the use of the following relational operators for searching numeric attributes:
 - greater than
 - greater than or equal to
 - less than
 - less than or equal to
 - equal to
 - not equal to
- d. ASV3 shall support the use of the Boolean operators "AND," "OR," and "NOT," including combinations of multiple Boolean operators, as a means of qualifying search criteria.
- e. ASV3 shall provide searching by pattern matching for text attributes including the optional use of one or more wild card characters in search criteria.
- f. Users shall be able to return to the most recently specified search criteria and edit them to refine the next search.
- g. ASV3 shall provide the capability to browse the catalog.
- h. Users shall be able to follow links between catalog entries to access entries for related objects.
- i. The users shall be able to determine their current location within the hierarchy.

- j. ASV3 shall provide the capability to view online objects.
- k. ASV3 shall support searching within online documents for the occurrence of a string of characters.
- l. ASV3 shall allow the addition of viewing tools for non-ASCII files.
- m. ASV3 shall support the viewing, printing, saving, and retrieving of search results.
- n. ASV3 shall provide easy to use file transfer applications for downloading objects from the library directly to the user's system or PC.
- o. A mechanism will exist to support queued printing of one or more documents and the copying of one or more objects to external media for physical distribution.
- p. Object queuing shall be supported while browsing an object and while examining lists of objects.
- q. ASV3 shall provide an electronic mail application supporting communications between all classes of AdaNET users. This application shall include the capabilities to create, edit, send, retrieve, acknowledge, delete, reply, and forward messages.
- r. ASV3 shall provide the capability of defining mailing lists of recipients of electronic mail messages.
- s. ASV3 shall support the distribution of electronic mail using complete network addresses. In addition, the users shall have the capability to define, update, and delete aliases for use in simplifying the addressing of electronic mail.

5.1.1.1.2 Maintenance Functions

The following requirements are for use of the operations staff and are not available to the general user community.

- a. ASV3 shall provide file transfer applications for uploading files to the ASV3 host machine.

- b. ASV3 shall allow the addition, archive, update, and deletion of objects.
- c. Changes to the catalog, whether by addition, modification, or deletion, shall be supported in a restricted mode, that is, the staff shall be able to enter all catalog changes then review and correct them as necessary prior to affecting the online system.
- d. ASV3 shall provide the same cataloging facilities for both online and offline objects whether the objects are stored locally at the repository or at a remote site.
- e. File preparation tools shall include, but not be limited to, the entry and editing of text and spell checking of text.
- f. ASV3 shall provide the following library statistical reports:
 - Registered Users by Number of Sessions
 - Objects accessed by collection
 - Objects Created/Modified/Archived/Deleted
 - Collections Created/Modified/Archived/Deleted
 - Accesses by collection (Quarterly and Annually)
 - Most Accesses/Requested Objects
 - Least Accesses/Requested Objects
- g. ASV3 shall provide the capability to collect data, measure performance, and support performance tuning of the system.

5.1.1.2 Security and Privacy Requirements

- a. ASV3 shall provide for the administrative creation of new user accounts and definition of user privileges.
- b. ASV3 shall provide log on and password capabilities as mechanisms for ensuring user privacy and the safeguarding of system files.
- c. ASV3 shall allow users to change their passwords.
- d. No other functions shall be available prior to successful log on.

- e. Connectivity to the ASV3 system shall be automatically terminated after three consecutive, unsuccessful log on attempts within one session.
- f. Inactivity on a user terminal for some period of time (to be determined) shall result in automatic log off of that user.
- g. ASV3 shall display a disclaimer message and receive user acknowledgement prior to presenting any options for service to the user.
- h. Library maintenance functions shall not be accessible by the general user.

5.1.1.3 AdaNET Holdings

- a. All information available in ASV2 at the time of transition to ASV3 shall be incorporated into the ASV3 system.
- b. The schema for the ASV3 catalog shall accommodate the most useful characteristics of objects. This includes, but is not limited to
 - Author
 - Organization, contact, address, and telephone number
 - Electronic contact mechanism and address
 - Title of product
 - Type (paper, course, conference, code, design document, etc.)
 - Abstract
 - Keywords
 - Adequate cross references and linkages to locate related components and documents such as documentation, source code, and object code as well as relevant commercial tools, papers, proceedings, reviews, conference announcements, and sources of training
- c. The schema for the catalog shall accommodate additional characteristics defined, as applicable, for each category.
 - 1. For courses and conferences, this includes, but is not limited to
 - Start and end date, delete after date

- Location and time
- 2. For software, this includes, but is not limited to
 - Date of development and/or date of last update
 - Size of online file and the number of lines of code
 - Location (online at AdaNET, offline at AdaNET, at author's organization only, etc.)
 - Software language
 - Target environment
 - Identification of hardware and operating system requirements
 - Operational restrictions (compiler, tools, peripherals, etc.)
 - Price
 - Restrictions on licensing, warranties, and liabilities
- 3. For documentation and publications, this includes, but is not limited to
 - Date of publication and/or date of last update
 - Size of online file and the number of pages, lines of code, etc.
 - Location (online at AdaNET, offline at AdaNET, at author's organization only, etc.)

5.1.1.4 Operational Requirements

- a. The AdaNET staff shall insure the maintainance of the hardware and all COTS software. The AdaNET development team shall support the software developed under their contract for an interim period of 60 days following the final acceptance of the ASV3 system.
- b. The AdaNET staff shall maintain user documentation in the form of getting started and quick reference guides distributable to new users and complete, online user documentation available for browsing and downloading.
- c. The AdaNET staff shall provide a help desk to assist users accessing the system.
- d. The AdaNET staff shall provide a help desk to assist users with information access.

- e. The AdaNET staff shall report user comments regarding their experiences and suggestions.
- f. The AdaNET staff shall accept and process information requests and orders.
- g. The AdaNET staff shall prepare objects for distribution in the form of printed copy, diskette, and magnetic tape.
- h. Methods of physical distribution shall include facsimile and U.S. mail or other common carrier, as appropriate to the medium and volume of information.
- i. The AdaNET staff shall accept and process problem reports.
- j. The AdaNET staff shall safeguard the contents of the library through application of appropriate backup policies, including placing the backup media in an off-site storage location.
- k. The qualification, cross referencing, and associating (linking) of the holdings in current and future baselines, shall be performed by the AdaNET staff.
- l. DELETED
- m. The AdaNET staff shall perform data archiving as related to maintaining library holdings.
- n. Journals (audit trails) shall be maintained in compliance with the AdaNET program configuration management procedures.
- o. The AdaNET staff shall perform configuration management and change control for both the operational repository software and the repository holdings.
- p. The AdaNET staff shall maintain a catalog of it's holdings which will be periodically published.
- q. The AdaNET staff shall maintain and publish information for users and developers working in specific domains of interest.

- r. The AdaNET staff shall provide, as resources permit, search services and queries to help the users locate components not in the cataloged repository.
- s. The AdaNET staff shall track component deliveries to users.
- t. When notified of defects in specific components, the AdaNET staff shall endeavor to notify those users who have received copies of the components of the existence of the defects as well as known fixes.
- u. The AdaNET staff shall retire unstable or otherwise defective components and, when possible, shall send notices of such retirement to any users that have previously received copies.
- v. The AdaNET staff shall provide value added services to any component listed in the catalog. At a minimum, these value added services include classification, qualifications, and cumulative confidence metrics.
- w. The AdaNET staff and RBSE team shall endeavor to work with appropriate contacts at all NASA centers to insure that this repository is the first choice for users interested in reusable products, processes, and interfaces developed on future NASA projects.

5.1.2 Host Computing System Requirements

5.1.2.1 Hardware Requirements¹

- a. The ASV3 host computer shall exhibit processing power of at least 20 SpecMarkS.
- b. The ASV3 host computer shall have a real memory capacity of at least 32 megabytes expandable to 128 megabytes.
- c. The ASV3 host computer shall have disk capacity of 2.5 gigabytes expandable to 6 gigabytes
- d. The disk storage shall be provided by multiple drives of at least 500 megabytes each.
- e. The average access time to the disks shall be no slower than 12 milliseconds.
- f. Terminals for AdaNET staff use shall include at least
 - 4 directly connected X-terminals for data entry
 - 2 directly connected Unix workstations capable of running X-Windows for library maintenance
 - 1 386 PC and 1 Macintosh which may access ASV3 through modems and which are capable of running X-Windows to be used for support of the ANSI terminal interface and for media generation for distribution
- g. The ASV3 host shall be capable of expanding to directly connect up to 16 terminals dedicated for staff use.
- h. ASV3 shall conform to IEEE 802.3 and ISO 8802/3 standards for Ethernet.

¹ Additional existing equipment is available for use in the ASV3 configuration if applicable. This includes two uninterruptible power supplies (a CPG 15 KVA 3-phase UPS and a Data General 10 KVA 3-phase UPS), a VitaLink TransLan 320 network bridge, and two NEC N500A/DSU modems for a dedicated DDS circuit, and a Data General 8-port transceiver. Existing multiplexors which can be interfaced to ASV3 include one NET (ComDesign) 64 channel SPX/50 MUX and one NET 8-link SPX/50 MUX.

- i. Existing modems for dial-up access which shall be interfaced to ASV3 include three Avatex 2400 baud modems, three Microcom AX/2400c modems. In addition, there is one Case 4096+ 9600 baud modem for access to SprintNet. These modems use RS-232C interfaces.
- j. ASV3 shall be expandable to connect up to 20 modems.
- k. An uninterruptible power supply (UPS) shall maintain system operations for at least 20 minutes in the absence of normal electrical power.
- l. Equipment shall exist to support the distribution of objects as follows:
 - printed copy
 - diskette (IBM PC 3.5 inch high or low density, IBM PC 5.25 inch high or low density, or Macintosh 3.5 inch)
 - magnetic tape (1/2 inch, 1600 bits per inch)
- m. A high-speed, high-volume backup device shall be provided which is capable of storing 1.5 gigabytes or more of data.
- n. Existing Data General laser printers (model # 4558 and # 4426) may be interfaced to ASV3, based on studies of the existing hardware.
- o. Deleted
- p. The ASV3 host computer shall have additional archival storage capacity of 2 GB to support the long-term storage of archived ASV3 holdings.
- q. ASV3 shall support remote, X window interface over a T1 class communication link.

5.1.2.2 Commercial Software Requirements

- a. ASV3 shall adopt an open systems architecture based on Unix and X-windows.
- b. Government-furnished and commercial-off-the-shelf (COTS) software shall be used as much as possible. Original

development shall be limited to those functions necessary to achieve the mission of the research program.

- c. The ASV3 platform must support the addition of an Ada compiler for eventual maintenance of software objects written in the Ada language.
- d. A compiler for the C language shall be provided to support maintenance of NELs and the development and maintenance of its tools.
- e. Text file preparation, maintenance, and spell checking shall be provided by COTS software.
- f. ASV3 shall provide word processing and publishing capabilities to directly connected UNIX workstations, ASV3 host connected IBM PCs and ASV3 host connected Macintoshes.
- g. The Transmission Control Protocol/Internet Protocol (TCP/IP), ISO 8473, shall be provided for network access.
- h. Network File System (NFS) server software shall be provided.
- i. File Transfer Protocol (FTP) software shall be provided .
- j. The Oracle database management system shall be used to support NELs.
- k. Subject indexing shall be accomplished with terms based on the AdaNET Thesaurus and other published thesauri or listings of terminology such as the IEEE Glossary of Software Engineering Terminology.
- l. A schema for the ASV3 catalog shall be provided, perhaps government furnished, which will allow appropriate and consistent cataloging of all the existing and near-term AdaNET holdings.
- m. A graphics terminal interface based upon the OSF Motif version of X-windows shall be used for all directly connected terminals.
- n. A COTS electronic mail utility shall be used.

- o. Both full and incremental disk storage backup capabilities shall be provided.

5.2 Performance and Quality Engineering Requirements

5.2.1 Capacity and Performance Requirements

- a. ASV3 shall be able to accommodate a library of 6000 to 8000 online objects expandable to 32,000.
- b. ASV3 shall provide acceptable response when loaded with 5 active users and 2 active librarians. Measurable response time requirements are as follows:
 - Average standard display time (full screen display time to call up menus, display help, or page a multiple screen display) not to exceed 1 second for a directly connected X-Window terminals.
 - Average standard display time not to exceed 8 seconds for ANSI terminals connected via 1200 baud modems.
 - Average keyword search and display of the first full screen of results not to exceed 5 seconds plus the standard display time for the terminal type.
 - The efficiency of file downloading over clean communications lines shall exceed 65% such that a 17,000 byte file can be downloaded at 1200 baud in less than 3 minutes.
- c. Spare capacity shall exist to concurrently handle up to 10 users and 4 librarians with no more than 50% degradation in response time.
- d. Time to perform partial screen updates for ANSI terminals should be minimized.
- e. DELETED.

5.2.2 Environmental Requirements

- a. All equipment shall operate normally at temperatures up to 75° Fahrenheit and relative humidity up to 50%.

- b. All equipment shall operate off of 120 VAC + 10% - 8% single phase, nominal 60 Hertz power.

5.2.3 Quality Assurance Requirements

- a. The ASV3 system and help desk services shall be available for normal use at least 99.5% of the time between 8:00 a.m. and 8:00 p.m. Eastern time, Monday through Friday, except Federal holidays.
- b. The ASV3 system shall be available at least 92% of the time during hours other than between 8:00 a.m. and 8:00 p.m. Eastern time, Monday through Friday.
- c. There shall be a hardware maintenance contract for the system which guarantees the preceding availability requirements are satisfied when averaged over any 30 day period.
- d. COTS software should be supported by the vendor during normal operating hours of the repository operations..
- e. The ASV3 program shall arrange technical support for government furnished equipment and software.

5.3 Safety Requirements

- a. All electrical installations shall conform to National Electrical Code and local codes and shall be maintained and altered only by qualified personnel.
- b. DELETED

5.4 Implementation Constraints

ASV3 shall be implemented on a Unix hardware platform supported by the COTS software necessary to host NELS. NELS will be furnished by NASA (see 5.1.2 for details).

AdaNET shall not produce an autonomous variation of NELS. Enhancements to NELS will be modularized and implemented so as to minimize the cost of

—
— maintenance and maximize the ability to accept updates from the baseline version.

—
— A primary concern is that the stated objectives are achieved. Beyond that the overriding concern is to produce ASV3 in a way that allows its functions to be extended and its capacity to be expanded. This is why ASV3 shall attempt to comply with the GOSIP FIPS 146. This will maximize its potential for accommodating ASV4 while minimizing hardware and software component replacement.

5.5 Site Adaptation

—
— The detailed description of and schedule for facility installation, installation planning, and database conversion will be provided in the ASV3 Transition Plan.

6.0 TRACEABILITY

TBD

7.0 PARTITIONING FOR PHASED DELIVERY

None. Except for any required maintenance releases, ASV3 will be delivered in a single step. Any future versions will be developed under separate specification.

8.0 ABBREVIATIONS AND ACRONYMS

ANSI	American National Standards Institute
ASCII	American National Standards Code for Information Interchange
ASV2	AdaNET Service Version 2
ASV3	AdaNET Service Version 3
ASV4	AdaNET Service Version 4
CCITT	International Telegraph and Telephone Consultative Committee
COTS	Commercial-off-the-shelf
FTP	File Transfer Protocol
GOSIP	Government Open Systems Interconnection Profile
IEEE	Institute of Electrical and Electronics Engineers
IBM	International Business Machines
ISO	International Standards Organization
MNP	Microcom Network Protocol
NASA	National Aeronautics and Space Administration
NFS	Network File System
OSF	Open Software Foundation
PC	Personal Computer
PMP	Program Management Plan
RFP	Request for Proposal
RFQ	Request for Quotation
RICIS	Research Institute for Computing and Information Systems
SMAP	Software Management and Assurance Program
TBD	To Be Determined
TCP/IP	Transmission Control Protocol/Internet Protocol
UPS	Uninterruptible power supply
VAC	Volts/alternating current

9.0 GLOSSARY

Alias	An alternate label.
Attribute	A characteristic; for example, attributes of objects include name, format, abstract, and publication date.
Availability	The degree to which a system or resource is ready when needed to process data.
Catalog	An ordered compilation of item descriptions and sufficient information to afford access to the items. An individual entry is sometimes referred to as metadata.
Collection	A group of related objects.
Electronic Mail	The electronic creation, modification, sending, answering, forwarding, manipulating, and receiving of messages between or among system users.
Ethernet	A coaxial cable network in which all stations monitor the network during their own transmission and terminate transmission if a collision is detected.
Kermit	A packet-oriented protocol developed at Columbia University and available on many different computer systems. By using a technique called 8th bit quoting, Kermit is able to transfer binary files between 7 and 8 bit systems.
Librarian	AdaNET personnel responsible for patron management and support, maintenance of the repository contents, catalog, and organization, as well as implementing AdaNET policies.
Object	A member of the library collection which may be on line (such as a program file, a data file, or a document file) or off line (such as a book).
Offline	An attribute ascribed to a service or function that is performed for a user in an asynchronous fashion (the user requests the service and continues performing a different function). The service is completed whenever resources are available.

Online	An attribute ascribed to any service or function that is performed for a user at an AdaNET-connected terminal in a synchronous fashion (the user cannot proceed until the service is complete).
Qualification	Verification of component compliance with AdaNET submission standards, policies, and procedures.
Reliability	The probability that a device, resource, or system will function without failure over a specified time period or amount of usage.
Response Time	The elapsed time between submission of an item of work to a computing system and the return of the results. For interactive terminals, the time between the end of user input and the display of the first character of the system response.
Reusable	The attribute that allows the same software to be used in two or more development efforts.
SpecMark	A measure of performance derived from the geometric mean of five integer and five floating point performance tests. The SpecMark rating of a machine is its performance relative to a VAX 11/780 machine. The benchmark programs are distributed by a consortium called SPEC.
SprintNet	A public data network operated by U.S. Sprint (formerly Telenet). SprintNet is an adaptive, flexible data network accommodating asynchronous terminal communication as well as interactive and transaction-oriented communication using protocols including, but not limited to, X.25.
Turnaround Time	The time elapsed from job submission until the job has executed and output is available. Normally referenced in connection with offline or batch jobs.
XMODEM	A block-oriented, error checking protocol released into the public domain by its creator, Ward Christensen.
YMODEM	A variant of XMODEM that supports longer data blocks (1K) and thus speeds transfer times.
ZMODEM	A variant of XMODEM that speeds transfer times.

10.0 APPENDIX A - CURRENT FUNCTIONALITY OF ASV2

	<u>Addressed in section</u>
Log on/off	5.1.1.2.b
User accounts	5.1.1.2.a
Password checking	5.1.1.2.b
Changing password	5.1.1.2.c
Catalog functions	
Public and private areas	5.1.1.1.2.c
Browse catalog information	5.1.1.1.1.f
Search	
By name for a specific document, folder, or drawer, i.e., for an object or a collection of objects	5.1.1.1.1.a
By keyword	5.1.1.1.1.a
By author	5.1.1.1.1.a
By date and time of creation or last update	5.1.1.1.1.a
By logical combinations of search criteria	5.1.1.1.1.c
By textual contents of document summaries	5.1.1.1.1.a
By textual contents of online documents	5.1.1.1.1.j
List management	
Save and print the results of searching operations	5.1.1.1.1.l
View documents	5.1.1.1.1.i
Private functions	
Edit documents and summaries	5.1.2.2.e
Update and delete collections and subcollections	5.1.1.1.2.b
Archive and restore collections and subcollections	5.1.1.1.2.b
Spelling checking	5.1.2.2.e
Functions restricted to librarians	
List active users	Unix function
Distribute information	
Floppy diskette	5.1.2.1.h
Magnetic tape	5.1.2.1.h
Printed	5.1.2.1.h
Electronic mail functions	
Read, save, reply to, forward, and send	5.1.1.1.1.p
Send a document.	5.1.1.1.1.q
Mailing list preparation and change	5.1.1.1.1.q
Define personal aliases	5.1.1.1.1.r
Support of TCP/IP mailing addresses	5.1.1.1.1.r
Download options via FTP or Kermit	4.2.b

Utilities

Define the personal profile for mail, word processing,
filing functions, language, and general
characteristics, i.e., customize various account
specific features including the handling of mail
establishing display options

5.1.1.2

Define and execute user commands

5.1.1.2

Miscellaneous

View the list of keywords

5.1.1.1.1.b

View the list of registered users

5.1.1.4

Four function calculator

omitted

11.0 APPENDIX B - ADANET's GROWTH FORECAST

Summary of AdaNET Catalog Sizing and Short-Term Growth - The report which follows shows that, based on the AdaNET Information Management Reports, the net number of cataloged, online objects in ASV2 increased by 393 over a 5-month period in late 1990 (an average of 79 objects per month from 2791 in June to 3184 in November). Extrapolation of this size and rate through the middle of 1993, when ASV3 is to be replaced by ASV4, forecasts a minimum ASV3 cataloging capacity of about 6000 online objects.

Estimates obtained from the data services personnel indicate that additions to the catalog average 300 per month while archival (removals) average 157. Thus, while not corroborated by statistical data, a forecast of a higher net rate of increase of 143 objects per month may be appropriate. This higher number indicates a catalog capacity of about 8000 objects by the middle of 1993.

ADANET's GROWTH FORECAST

November 23, 1990

PURPOSE

The purpose of this report is to describe the size of the current online AdaNET database and forecast its future rate of growth.

Karen Fleming from MountainNet commented that AdaNET's rate of growth varies from month to month, and she could not give an estimate nor could she project the size of ASV3 or ASV4 before receiving the project's requirements information.

Below is information about the AdaNET holdings including both the online ASV2 documents and the hardcopy documents filed in the AdaNET Library located at MountainNet in West Virginia. The offline AdaNET Library is a project support library that is cataloged separately from the online system.

Information Management Report for June 1990

There is a total of 2791 documents online ASV2, and the external AdaNET Library at MountainNet consists of 1196 documents. Ninety-Six (96) documents were cataloged, processed, and added to the AdaNET Library during this report period.

Information Management Report for November 1990

There is a total of 3184 documents online ASV2, and the external AdaNET Library at MountainNet consists of 1361 documents. Seventy-one (71) documents were cataloged, processed, and added to the AdaNET Library during this report period.

The following graph illustrates two different AdaNET growth projections. The first one is calculated using Information for Business's (IFB) monthly averages. Rebecca Bills from Information for Business reports that a monthly average of 300 documents added to the online ASV2 Library, and Joyce Combs from Information for Business reports that an average of 157 documents are archived each month. These figures do not include reusable software. Therefore, there is a net increase of 143 documents per month or 1716 documents per year. The second one is calculated by taking the difference between the number of online documents listed in the June and November Status Reports and dividing by five months ($393/5$) to get a monthly average. Therefore, there is an increase of 79 documents per month or 948 documents per year.

At this rate of increase (143 documents per month), projections for AdaNET's growth are:

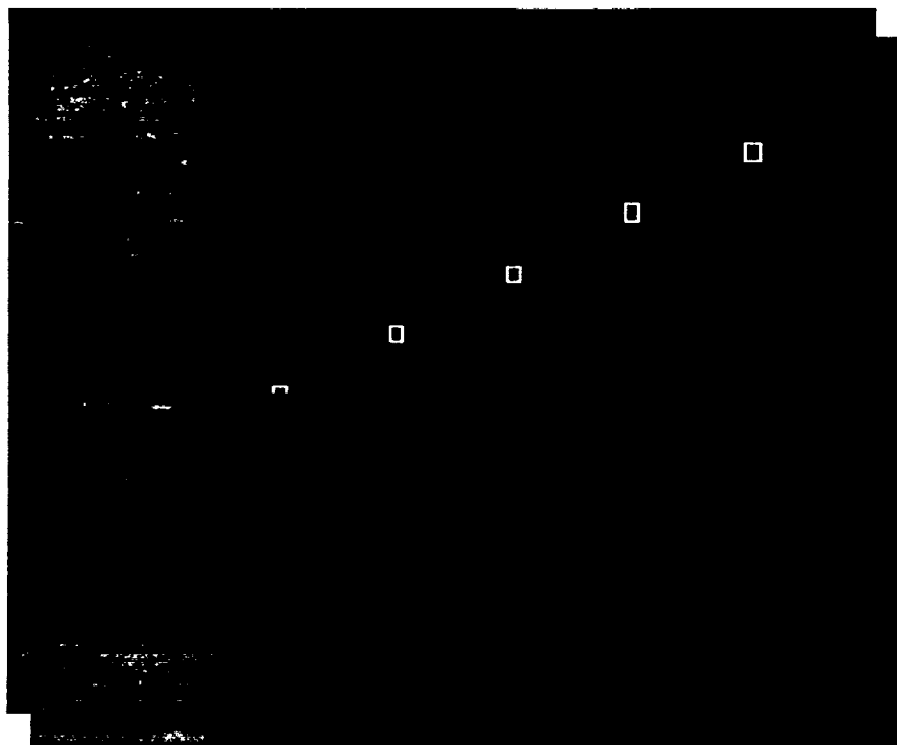
July 1990 -	2934 documents online
January 1991 -	3470 documents online
January 1992 -	5186 documents online
January 1993 -	6902 documents online
January 1994 -	8618 documents online
January 1995 -	10,334 documents online

At this rate of increase (79 documents per month), projections for AdaNET's growth are:

July 1990 -	2870 documents online
January 1991 -	3344 documents online
January 1992 -	4292 documents online
January 1993 -	5240 documents online
January 1994 -	6188 documents online
January 1995 -	7136 documents online

Assumptions: These calculations assume that the number of documents added and archived will remain constant throughout this time frame.

ONLINE ADANET DOCUMENTS



ORIGINAL PAGE IS
OF POOR QUALITY

APPENDIX F
DELIVERABLES COVER SHEET

RB.05 GHG Corporation
January 1, 1992

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW #6 (Six)

Title of Task: Online Information user's survey report - draft

Subcontractor: GHG Corporation

Cooperative Agreement No. NCC-9-16

Principal Investigator: E.T. DICKERSON

NASA Technical Monitor: E. FRIDGE

Type of Report: _____

Period Covered by Report _____

Due Date: 03/23/92

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

GHG Corporation
1300 Hercules, Suite 111
Houston, Texas, 77058

23 March 1992

Mr. Gary Hamel
RICIS Document Control Center
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas, 77058-1096

Dear Sir,

Enclosed is a hard copy of the Draft Online Information User's Survey Report for the Repository Based Software Engineering (RBSE) Program.

Should you have any questions or if I can be of further service, please don't hesitate to call at (713) 488-8806. Thank you for your time and effort.

Sincerely,



Robert C. Frederiksen
Configuration/Data Management

cc: E. T. Dickerson

**Online Information User's
Survey Report DRAFT**

**Repository Based Software
Engineering (RBSE) Program**

March 1992

**Cooperative Agreement NCC9-16
Project No. RICIS No. SE.18**

**Submitted to:
University of Houston - Clear Lake
2700 Bay Area Boulevard
Houston, TX 77058-1096**

**Submitted by:
GHG Corporation
1300 Hercules, Suite 111
Houston, TX 77058**

**Online Information User's
Survey Report DRAFT**

**Repository Based Software
Engineering (RBSE) Program**

March 1992

PREPARED BY:

A handwritten signature in cursive script, reading "Colleen P. Lippert", is written over a horizontal line.

Colleen P. Lippert
GHG Corporation

Online Information User's Survey Report DRAFT

December 2, 1991

Colleen P. Lippert

PURPOSE

An Online Information User's Survey was conducted at the 1991 Tri-Ada Conference during the week of October 21-25. An online information system is defined as any type of electronic card catalog, software repository, database service, or reservation system, etc. The purpose of the attached questionnaire is to gather information from online information system users to develop requirements for ASV4. Through this survey, the target market provides specific data about what users want and need in an online information system.

The target market includes anyone who uses or ever used an online information system. Since most online information system users attend the annual Tri-Ada Conference, it was determined that this conference would provide a sizeable convenience sample that is representative of typical users. The assumption is that the users who attend the Tri-Ada Conference are typical of all online information system users.

A total of 102 questionnaires were completed. However, only actual users are qualified respondents. Therefore, the 9 questionnaires that reflected nonusers (those who NEVER logged on to an online information system) are eliminated, resulting in a sample size of 93.

Not all of the questions have been answered by every respondent. Therefore, each question will be analyzed separately according to the number of responses.

Online Information Issues

This section provides data about the types of online information systems that are currently available, the demand for them, and user's attitudes about them.

1. a. Which type of online information system (s) do you use or have you used?

The first question was completed by 90 respondents. Figure 1 illustrates the types of online information systems used now or ever used by these respondents. Since respondents may use or have used more than one system, there are more than 90 answers to this question. The system used by most respondents is CompuServe with 43% of the users. AdaNET has 31% of the users. Only 1% of the respondents use US Videotel. Other

types of bulletin boards, networks, and private inhouse systems are used by 28% of the respondents, and those specified by users are listed below.

ABI/Inform	DDN/DSN	NARDAC
AdaIC	EDN	Navy
America Online	GRACE	Netnews
Apple Bulletin Board	IBM Bulletin Board	STSC
ASM	Internet	USA Today
BIX	MINITEL	Usenet

b. Do you (or your company) pay for this service?

There are 52 users who pay (or their company pays) for the systems and 44 users who do not. Since some respondents use more than one system, there are more than 90 answers to this question. Since CompuServe has the most users, it is valid that there are more users or companies who pay for this service. However, since cost was mentioned as one reason why certain systems are used more than others, it is a major user concern.

c. Which system (s) do you use the most and why?

<u>System</u>	<u>Reason Used Most Often</u>
ABI/Inform	free
AdaIC	easy access over Internet
AdaNet	source of information, type of information
America Online	Mac-oriented
BIX	high tech content, flat rate
CAMP	best tools
CompuServe	convenient, library search, has data format, availability of information, availability to discuss in forums, type of software
GENie	familiarity

MINITEL	many available services
Netnews	high tech content
PRODIGY	price, no hourly fee, easy to use, most service for little money
RAPID	required by contract
STARS	best contents, mandated

d. Which system (s) do you like the most and why?

System	Reason Liked Most
ABI/Inform	free
AdaNET	ease of use, diversity of information
ASR	convenient
BIX	high tech content, flat rate
CompuServe	variety of information, most useful libraries and forums, most flexible
GENIE	low cost, versatility, depth
MINTEL	many available services
Netnews	high tech content
STARS	free
PRODIGY	most service for the money, user interface
RAPID	reliability search criterion
Usenet	friendly
USA Today	reasonable cost

e. Which system (s) do you like least and why?

<u>System</u>	<u>Reason Liked Least</u>
AdaNET	incredibly hostile to PC users, not workable, information not easy to locate, cumbersome, have not been able to connect due to timeouts
America Online	least familiar
ASR	lack of testing, difficult to find useful information, no library system, hard to logon, hard to access and retrieve from, not enough access methods, no browse/retrieval
CIS	cost too much
CompuServe	hard to navigate, not cost-effective, expensive
PRODIGY	policies, no privacy, too slow
RAPID	lack of flexibility and query ability
STARS	format, difficult to find useful information

f. What improvements would you like to see in these systems?

Users suggested improvements for online information systems, and they are grouped into the following categories:

Access Methods and Format

ease of access

ease of use to understand menus and instructions/flatter menus

more remote access methods

very simple format and client server architecture

Additional Functionality and Services

more flexibility for other types of machines
more reuse lookup tools
more Ada
more training

Cost

less online cost/reduce costs

International Access

availability in Canada

Saving Methods

capability to save

Search Methods

better searching methods/multiple search strategies
better cataloging and cross referencing
better indexing review of software
more free form unstructured query capability for components

Security

minimal intrusion on user's privacy

Speed

speed it up

User Interface

better user interface/friendly window interfaces/hypertext interface
more graphical interfaces
having common interfaces for the different services

2. How often have you logged on to an online information system?

Respondents who answered None were disqualified from the survey process, since our target market consists of online information system users only. This question was completed by 86 respondents who logged on to a system at least once. Of these respondents, 40% of the users logged on

frequently, and 55% logged on occasionally. Only 5% of the users logged on to a system once. The majority (95%) of our survey respondents logged on to an online information system frequently or occasionally. Figure 2 illustrates how often users logon to an online information systems, and it also represents the demand for these systems.

3. a. How do you logon to these systems?

There are 86 respondents who answered this question. However, there are a total of 106 responses since users logon to several systems using different methods. Of these users, there are 69% who logon indirectly through a modem. Networks are used by 34% of the respondents. Some of the networks listed are Internet, Ethernet, Telenet, Usenet, ARRA, Bitnet, Lan, Milnet, and PC-NFS. Other local networks are used by 2% of the respondents, whereas 19% of the respondents are directly connected to a terminal. The most popular logon methods are modems and networks among the majority of the users as illustrated in Figure 3.

b. What type of terminal (s) do you use to logon?

There are 90 respondents who answered this question, but there are 116 answers due to users having access to more than one terminal. PC terminals are used by 72% of these respondents. Macintosh and VTx100 terminals are each used by 21% and 20% of the respondents respectively. The remaining 16% of the respondents specified other terminals. These include X windows, Sun Unix Workstation, Sun 4, Sun Sparstation, Amiga 2000, Minitel Device, and Rational. Figure 4 illustrates that the majority of respondents have access to PC terminals. The two types of PC CPU classes mentioned are 286 and 386, and the Mac CPU class specified is IICI.

c. Which communications packages have you used?

This question was completed by 83 respondents. However, there are a total of 137 answers because respondents may have used more than one communications package. Kermit is used by 72% of these respondents, and 52% of the respondents use ProComm or ProComm Plus. Other types of communications packages are used by 40% of the respondents. Some of these are Versatarn, Red Ryder, CIM, Profs, PCU, BLAST, White Knight, MacKnowledge, PCTalk, Smartcom, TCPIP, Em4010, XTalk, TAPCIS, NFS, Unicom, and UUCP. Figure 5 illustrates that the communications package that is used the most is Kermit with ProComm or ProComm Plus as the second.

4. Which wide area networks are available for your use?

There are 83 respondents who answered this question. However, there are a total of 102 answers because users may have more than one wide area network available to them. Internet is used by 58% of these respondents, and 36% of the respondents use Telenet/Sprintnet. Other networks such as Milnet, DDN, Decnet, NSFnet, and PSCN are used by 10% users. There are 19% users without any wide area networks available to them. Figure 6 illustrates that Internet is the most available wide area network with Telenet/Sprintnet as the second.

5. Please rate each feature/benefit on the basis of its importance to you (1 being the least important and 5 being the most important).

There are 83 respondents who answered this question. The mode method of measurement is used to analyze the importance of nine system features/benefits to users. The Bulletin Board is rated 1 (least important) and 5 (most important) by the same number of users. Electronic Mail is rated 5 (most important) more than any other feature, and speed is rated the second most important benefit on these online information systems.

Figure 7 illustrates the least and most important system features/benefits rated by online information system users. Other important features or benefits that users specified are FTP download, detailed catalog, quality search, and availability of online source code.

Users also rated how they want information from an online information system shipped to them. Users rated diskettes as the most important way to ship information. Diskette sizes specified are 3.5, 5.25, and 1.4 PC-DOS. Users rated cassette tapes as the least important form of shipping information.

User Interface Issues

This section provides data about users' experience with various user interfaces and which user interface features are important to them.

1. What is your experience level with the following interfaces (1 being familiar and 5 being expert)?

Users indicated how experienced they are with five different user interfaces. These user interfaces are Macintosh, DOS, Microsoft Windows, X Window, and UNIX. Online information system users are least experienced with X Windows and most experienced or expert with DOS. Therefore, if users are more comfortable with an interface that they are more familiar with, then they prefer a DOS interface. Other interfaces

listed by users are Common Windows, IBM's TSO/ISPF, VAX/VMS, Interviews, Motif, and Amiga OS. Figure 8 illustrates the users' experience levels with various interfaces.

2. Please rate each user interface feature on the basis of its importance to you (1 being the least important and 5 being the most important).

Users rated the importance of nine different user interface features. The mode method of measurement is used to analyze the importance of specific interface features to users. The Mouse and Command Line are rated 1 (least important) more than any other feature, and the Keyboard is rated 5 (most important) more than any other feature. Figure 9 illustrates the importance of various interface features to users. Other features listed by users are Trackball, Undo Key, Scripting/Macros, Hypertext, Replay, and an Undo/Stop key.

Software Reuse Issues

This section provides data about the demand for reusable software and the potential market for it.

1. How often have you reused a software component?

This question was completed by 93 respondents. The result is 55% of users who reuse software components occasionally, and 33% who reuse software components frequently. Only 9% of the respondents never reused a software component. Figure 10 illustrates that a majority of online information users are reusing software components.

2.
 - a. How often have you accessed an online information system for software components to reuse?
 - b. Which system (s)?

There are 93 respondents who answered this question. Online information systems are accessed occasionally by 42% of the users for software components to reuse, and 18% of the users access a system for reusable components frequently. Since the majority of users access online information systems for reusable components, this indicates the demand for online reusable software components. Some of these systems are listed below.

AdaNET
AdaIC
ASR
Bitnet

BIX
CompuServe
Dec
NARDAC

Rational
RAPID
STARS
Usenet

Another 33% of the users indicated that they never accessed an online information system for reusable software components. Figure 11 illustrates how often online information systems are accessed to reuse software components.

3. a. **How often have you accessed an online information system to purchase software components and/or libraries of components to use?**
 b. **Which system (s)?**

This question was completed by 92 respondents. Online information systems are accessed frequently to purchase software components by only 3% of the users and occasionally by 21% of the users. Some of the systems that have been accessed for this purpose are CompuServe, GRACE, local BBS, and STARS. Figure 12 illustrates that the majority of users (70%) have never accessed an online information system to purchase software components and/or libraries of components to use.

4. a. **How often have you purchased software source code for reuse?**

There are 91 respondents who answered this question. Only 4% of the respondents purchased software source code for reuse frequently. Reusable software source code was purchased occasionally by 27% of the respondents. Figure 13 illustrates that the majority (62%) of the respondents never purchased software source code for reuse.

- b. **Approximately how many functions or procedures have you purchased?**

There are 35 respondents who answered this question. One to ten functions or procedures have been purchased by 51% of these respondents, and 19% of the respondents have purchased 11-50 of them. More than 100 functions or procedures have been purchased by 22% of the respondents. The majority (70%) of the respondents have purchased between 1-50 functions or procedures.

- c. **Please mark the types of reusable software source code which you have purchased.**

There are 44 respondent who purchased reusable software source code. Of these users, 68% of them purchased libraries of utility functions. Another 41% of the users purchased programming tools, 36% purchased libraries of complex functions, 32% purchased add-on packages, and 27% purchased application programs. Only 9% of the users purchased system programs.

Other types these users purchased are tools, datafiles, GIFs, Operating Systems, and Graphic Libs.

5. Which of the following would you (or your company) be interested in and/or would purchase from an online system?

There are 72 respondents to this question. The list of online information includes software requirements documents, software designs, software source code, software operational support documentation, verification and test suites, executable files only, user comments on experiences, discrepancy reports, published standards, technical papers and/or processes, newsletters, catalogs of software repositories, commercial software products, product evaluations and reviews, software engineering bibliographies, training sources/conference announcements, and business opportunities. Of these respondents, 65% are interested in newsletters, and 57% are interested in software source code.

However, many of these same respondents would not purchase this information. There are 39% of these respondents who would purchase software source code, and 26% would purchase commercial software products. Even though there is a high interest (65%) in newsletters, only 17% would actually purchase them from an online system. It is apparent that users are interested in many different types of online information (such as software designs) when they are offered free of charge, however, they are not interested in purchasing them.

Search Issues

This section provides data about what search mechanisms are important to users and what type of information service would be best for them.

1. Please rate the following search mechanisms based on their importance to you when searching for software information (1 being the least important and 5 being the most important).

There are 78 users who responded to this question. The mode method of measurement is used to analyze the importance of search mechanisms to users. Searching by keyword is rated 5 (most important) more than any other feature. However, searching by author is rated 1 (least important) and 5 (most important) by the same number of users. Figure 14 illustrates how users rate search mechanisms. Other search mechanisms that users specified are faceted taxonomy, application domain, and performance characteristics.

2. Which one of the following would be the MOST helpful to you in your work?

There are 81 respondents who answered this question. The choices are library service, catalog service, online repository system, and library management framework. Of those users, 73% selected the online repository system. Library service was chosen by 21% of these respondents, and library management was selected by 14% of the users. Only 10% of the users thought that a catalog service would be the most helpful. Another suggestion by one user was to provide a catalog where users could search and order online. Figure 15 illustrates the high demand for online information systems.

User Issues

This section provides general information about the user.

1. What language (s) do you use to program?

There are 80 users who responded to this question. Of these respondents, 94% of them use Ada, and 44% use C. Fortran is used by 18% of the respondents, and C++, and Pascal are each used by 15% of the respondents. Other programming languages mentioned are Lisp, Hypertalk, SQL, Assembler, Perl, Awk, Icon, Cobol, Basic, FTN, Modula, Clario, and Unix shells.

2. Which of the following file transfer protocols would you prefer to use for downloading information?

There are 86 respondents who answered this question. Users had to choose from Kermit, YMODEM, File Transfer Protocol, Network File System, uucp, XMODEM, and ZMODEM. Kermit is preferred by 50% of the users, and 44% of the users preferred File Transfer Protocol (FTP from OSI). Another file transfer protocol specified is BLAST. Figure 16 illustrates the file transfer protocols that users prefer.

3. Please indicate the hours that you would be MOST LIKELY to use an online information system. (Please indicate AM and PM).

There are 62 respondents who answered this question. Of these users, 45% of them specified working hours (8am - 5pm) as the time that they would most likely use an online information system. There are 5% of the users who logon before 8 am, and 24% who logon after 5 pm. There are 13% of the users who logon sometime before 8 am to sometime before 5 pm, 10% who logon from sometime after 8 am to sometime after 5 pm, and 3% who logon sometime before 8 am to sometime after 5 pm. Therefore, the

majority (55%) of the users would be using the system sometime before 8 am and/or sometime after 5 pm.

4. a. Do you have information that would be useful to submit to a software engineering repository?

There are 89 users who answered this question. Of these respondents, 78% do not have information to submit to a software engineering repository, while 22% of these users do.

b. If yes, briefly describe.

Some of the information that users would submit to a software engineering repository is listed below.

- MS Pascal based form/menu system
- Ada SQL Bindings
- GUI (XWindows) Bindings
- Reusable Components
- CAD Algorithms
- Ada Packages
- Ada Bindings to ORACLE and also EzX and TAE X Window tools
- Decimal authentic and other Ada Interfaces
- Trade Studies
- Standards Assessment
- Generic Technical Architecture
- Abstracts for papers on software engineering topics
- Tool Kits
- AGORA User Interface
- Management System
- Some ASR Modifications
- New Sources of Information

c. If yes, would you be willing to submit it?

Of those users who have information to submit to a software engineering repository, 80% would be willing to submit it. Only 10% would not submit information, 10% did not respond. One reason mentioned by users for not submitting information is that their company will not allow it because their work is proprietary and cannot be released to the public.

d. If yes, would you be willing to support it?

Of those users who would submit information to a software engineering repository, 60% of them would be willing to support it. Of those users

who would be willing to support it, 25% would support this information free of charge, and 35% would support this information for a fee.

Demographic Issues

The answers to these demographic questions provide descriptive statistics about the characteristics of online information system users.

1. Please state your job title and a brief job description.

These online information system users have many different job titles. They can be divided into the categories of Management (President, Director, Manager, Chief), Technical (Analyst, Engineer, Programmer, Scientist, and Specialist), Academic (Teacher), and Others (Consultant, Captain, Officer, and Contract Monitor). The majority (60%) of online information users are involved in technical work; whereas 23% are involved in management positions. Only 3% are involved in academic areas. Another 13% have various independent jobs such as consulting or government officer positions.

2. How would you classify your organization?

This question was answered by 90 users. Of these users, 43% are government contractor employees, and 32% are government employees. Private industry employees consist of 21% of the users, and only 3% of the users are from the academic world. Figure 17 illustrates that the majority (75%) of current online information users are government or government contractor employees.

3. To which professional organizations do you belong?

This question was answered by 60 respondents. Most users belong to more than one organization. The organizations that most users belong to are IEEE and ACM. Over 60% of the users belong to each organization. Other organizations that users specified are AFCFA, AAS, AIAA, AAAI, AFA, ACS, BCS, CPSR, CSI, CS, ISAC, ICCA, SIGAda, SCS, and SLA.

4. What professional conferences do you attend?

This question was answered by 64 users. Many of the online information system users attend more than one conference. Most of these users (78%) attend the Tri-Ada conference. The SIGAda conference is attended by 13% of the users, and 11% of the users attend the WADAS conference. Other conferences specified by the users are: AAAI, ACM, Ada Europe, AdaJUG, AdaSIG, ADUS, AFCEA, ASEET, ASWEZ, DECUS, FCC, FRAWG, ICSE,

IEEE Services, ISAC, Networld, NSIA, OOPSLA, SDE, SIGCHI, SIGGRAPH, SIGPLAN, and XHIBITION.

CONCLUSION

The majority of respondents to this online information system user's survey have accounts with CompuServe. Reasons such as forum discussion and availability of information are stated by users to explain why they use it and like it more than any other system. Cost is the main reason that CompuServe is not liked. Therefore, the requirements for ASV4 should include those features and benefits that CompuServe offers, and it should continue to be a free service. Adding these extra features and benefits to our free service will not only improve the system, but it will attract more users.

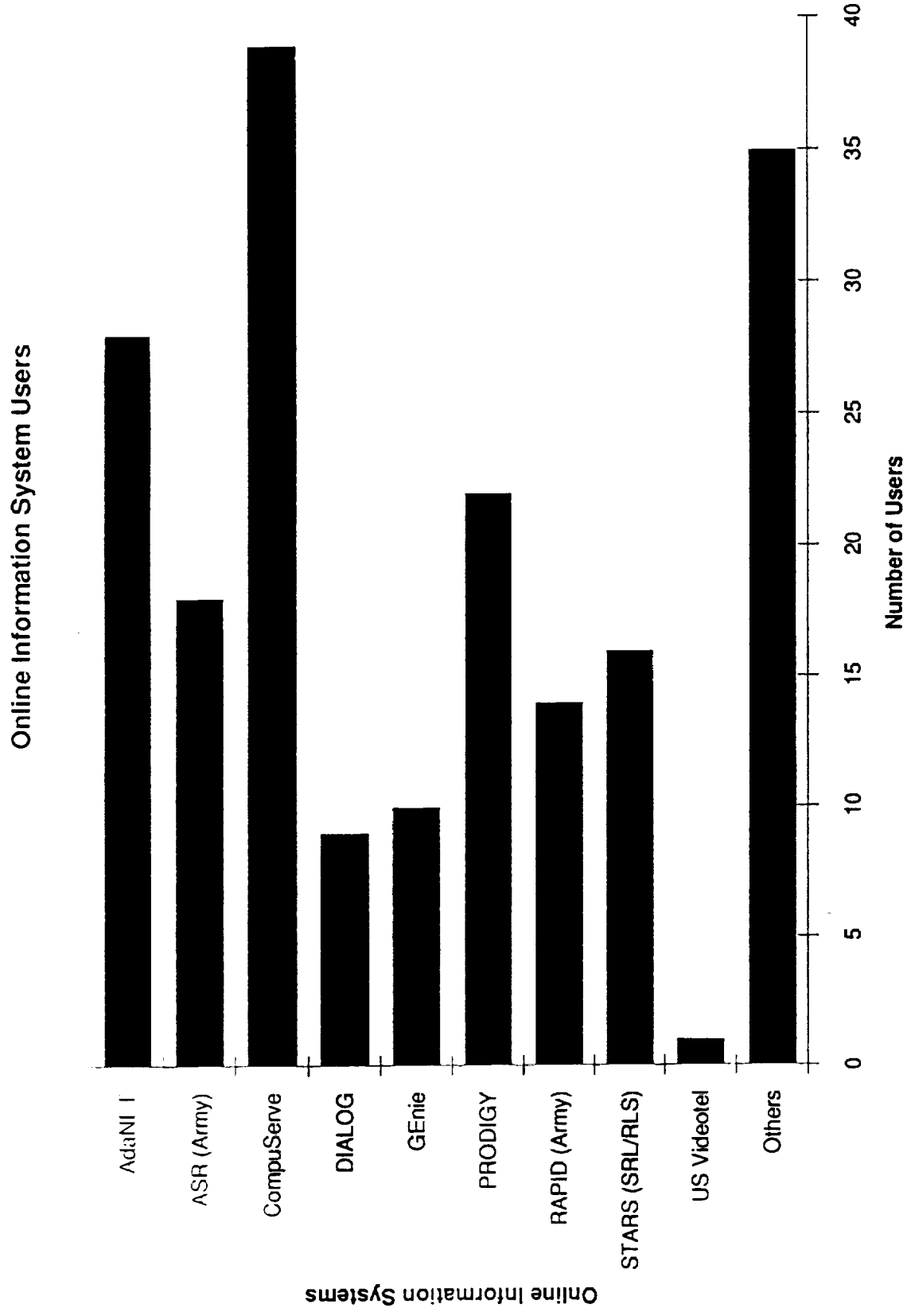


Figure 1- Online Information Issues

Online Information System User's Logon Frequency

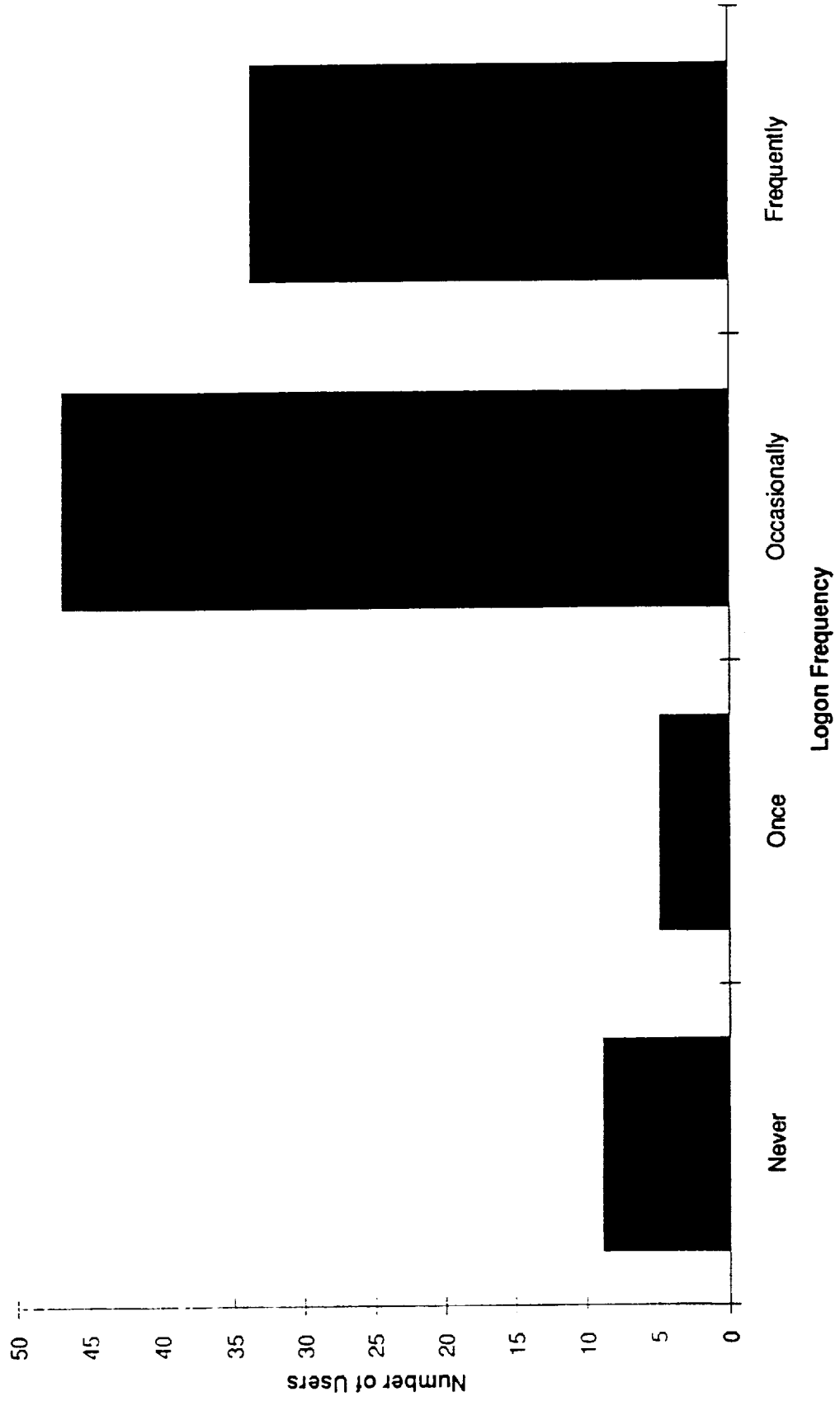


Figure 2 - Online Information Issues

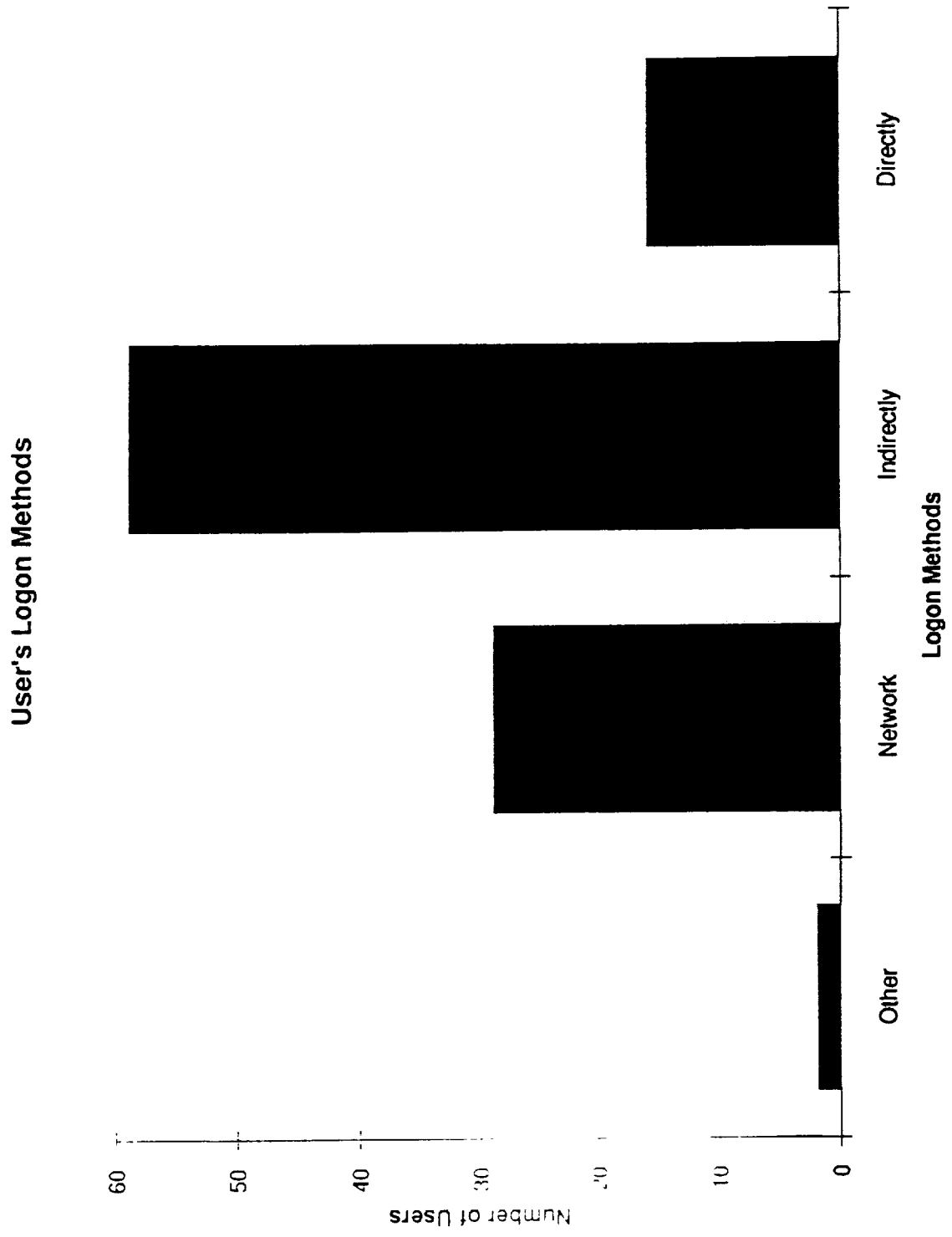


Figure 3 - Online Information Issues

Terminals Used to Logon

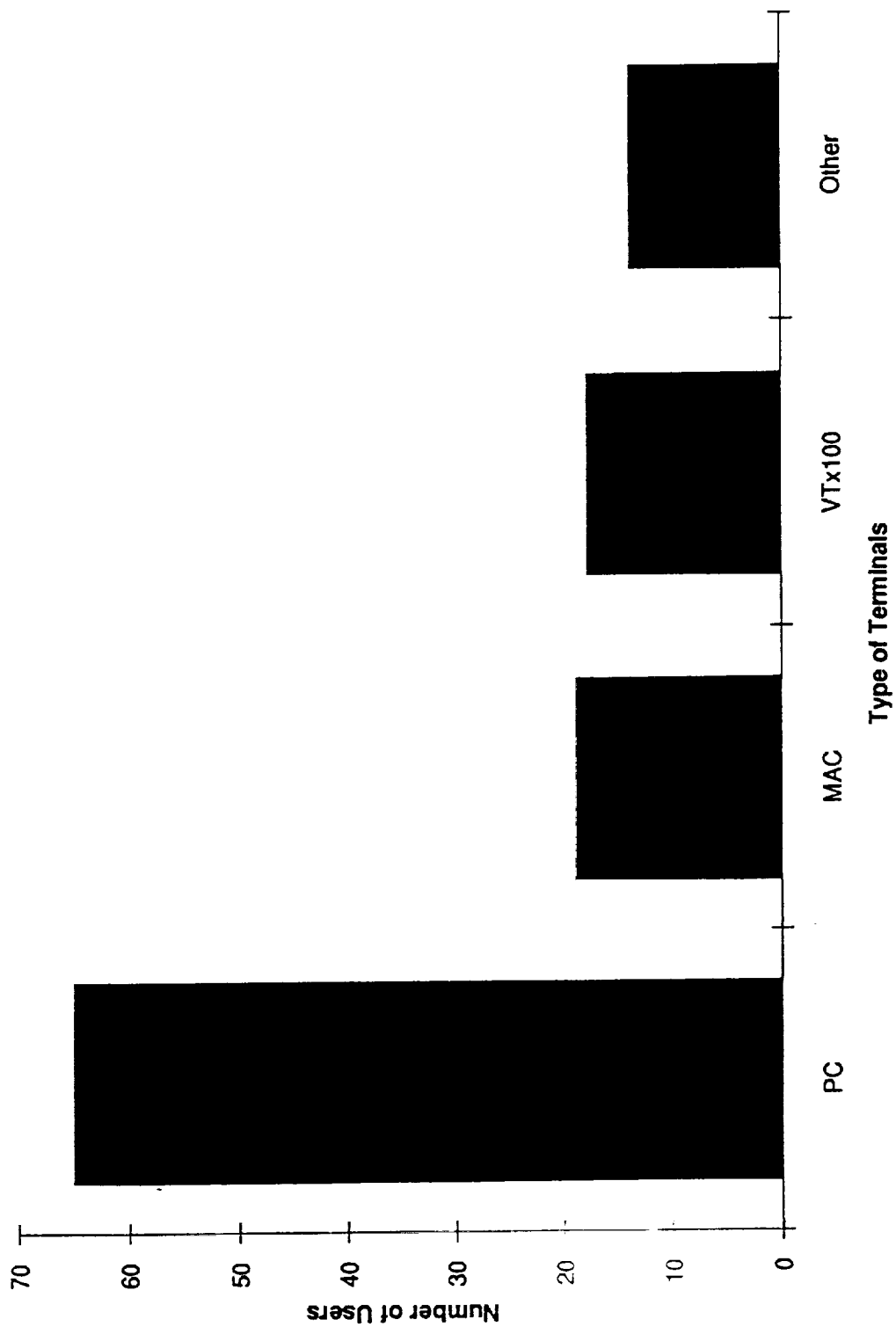


Figure 4 - Online Information Issues

User's Communications Packages

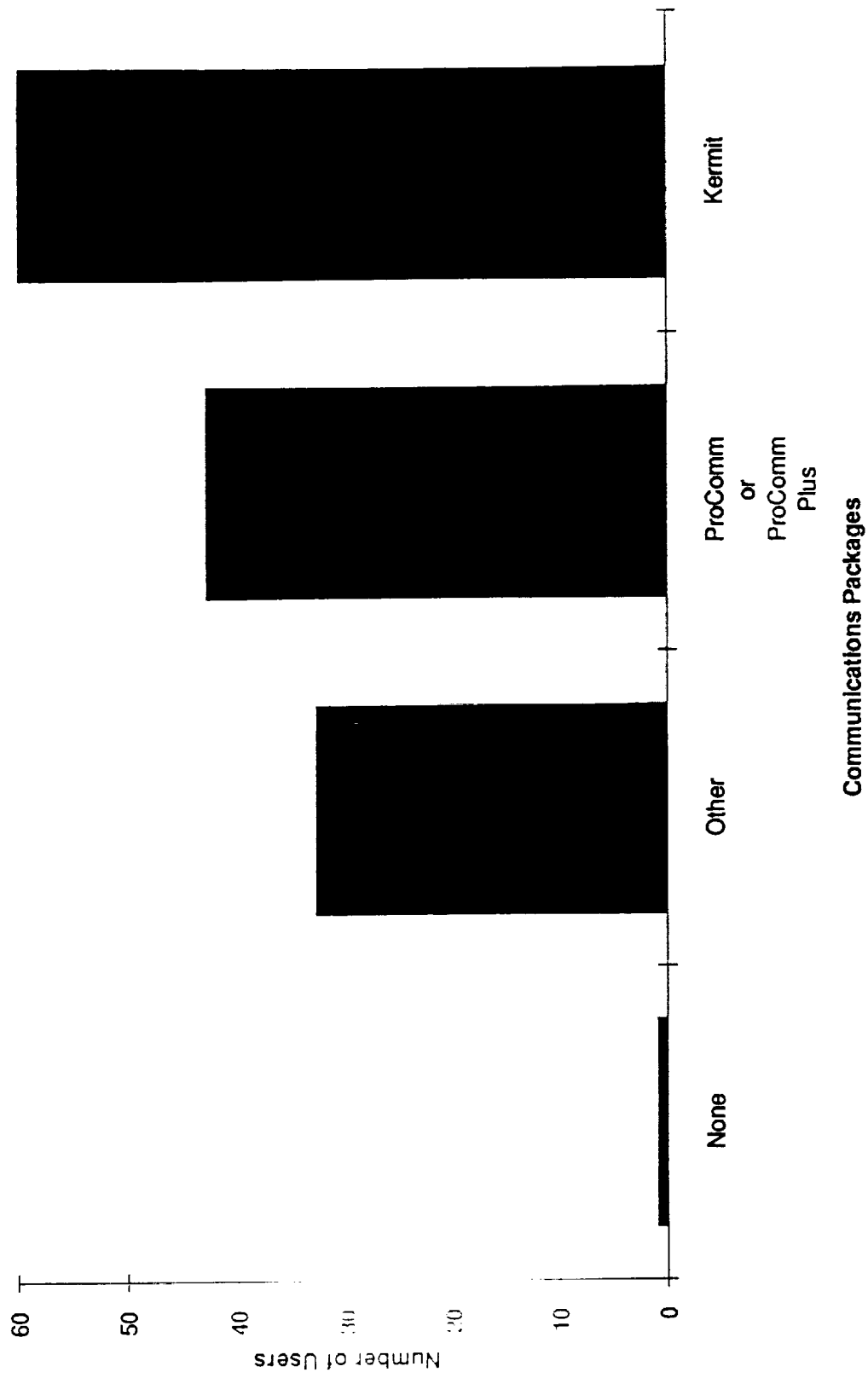


Figure 5 - Online Information Issues

User's Wide Area Networks

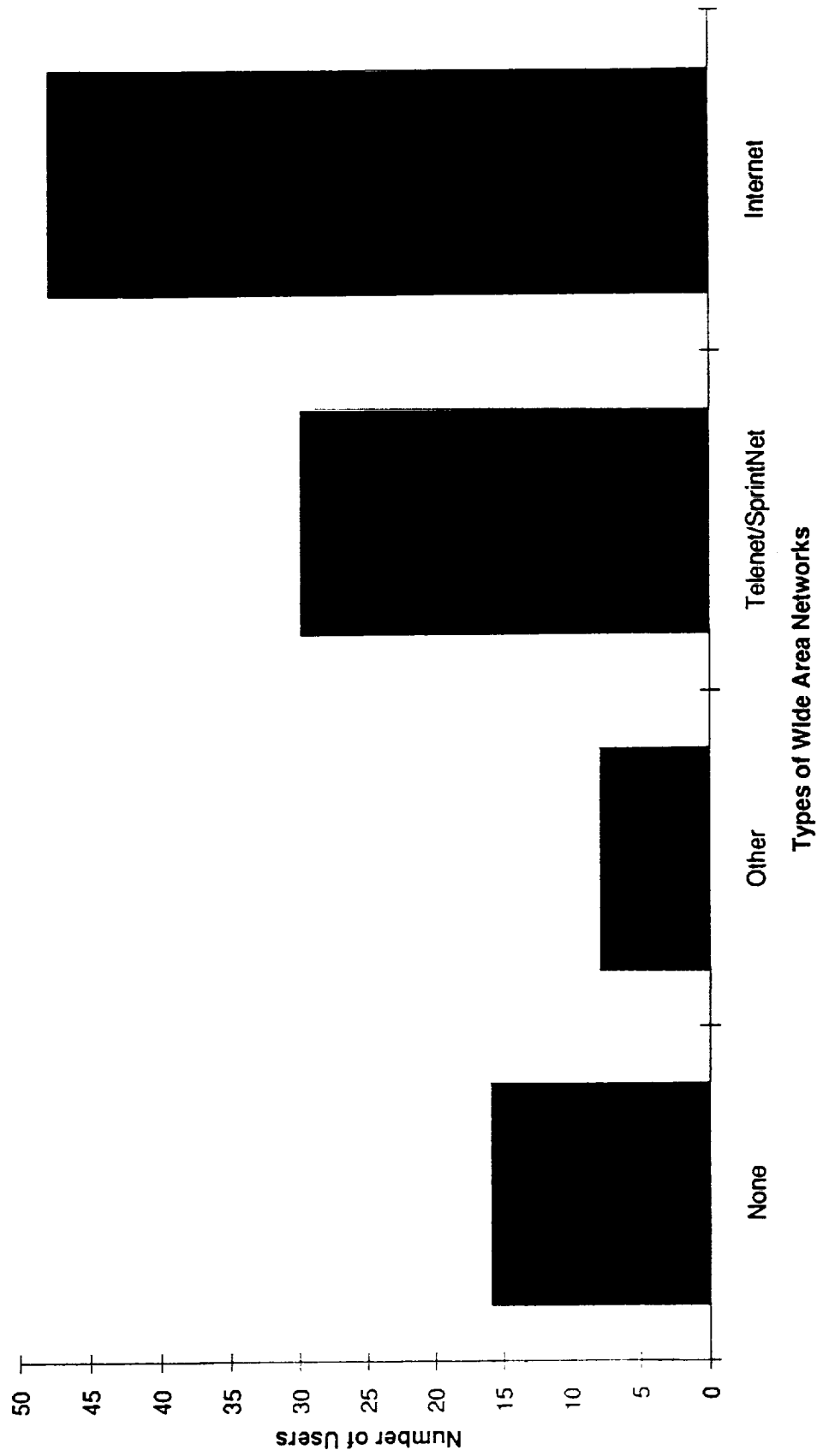


Figure 6 - Online Information Issues

System Features/Benefits Rated Least/Most Important

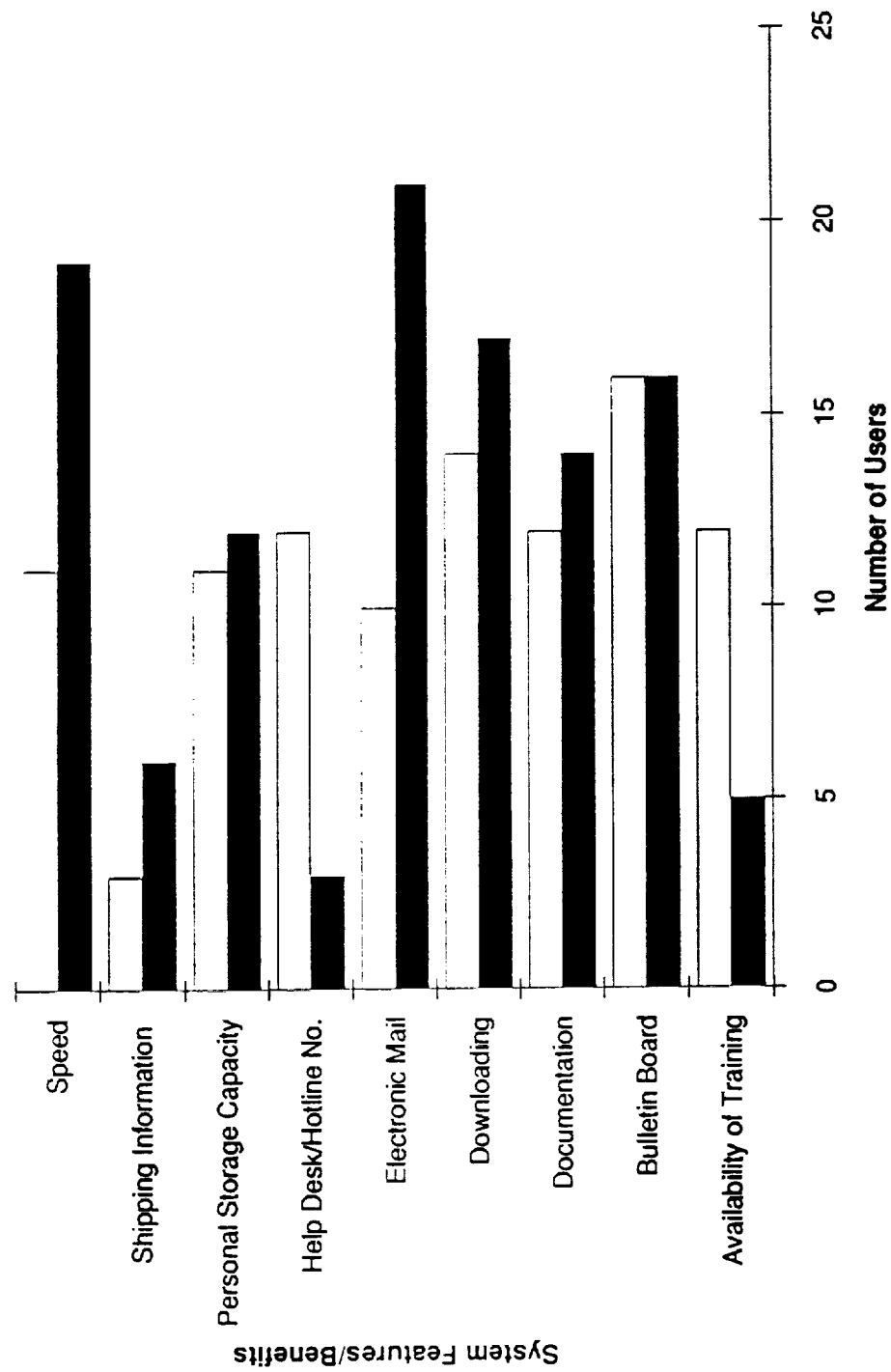


Figure 7 - Online Information Issues

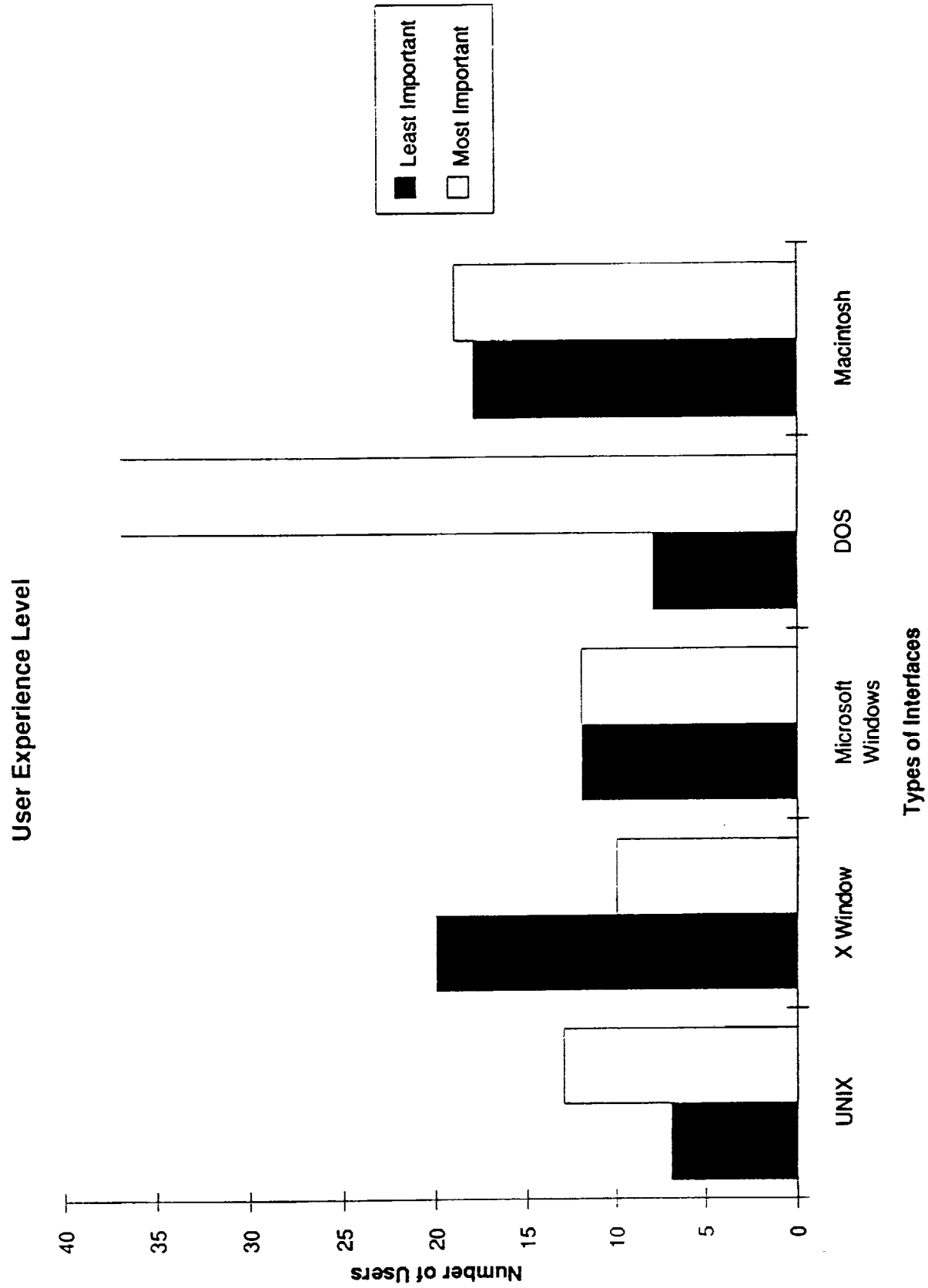


Figure 8 - User Interface Issues

Least/Most Important User Interface Features

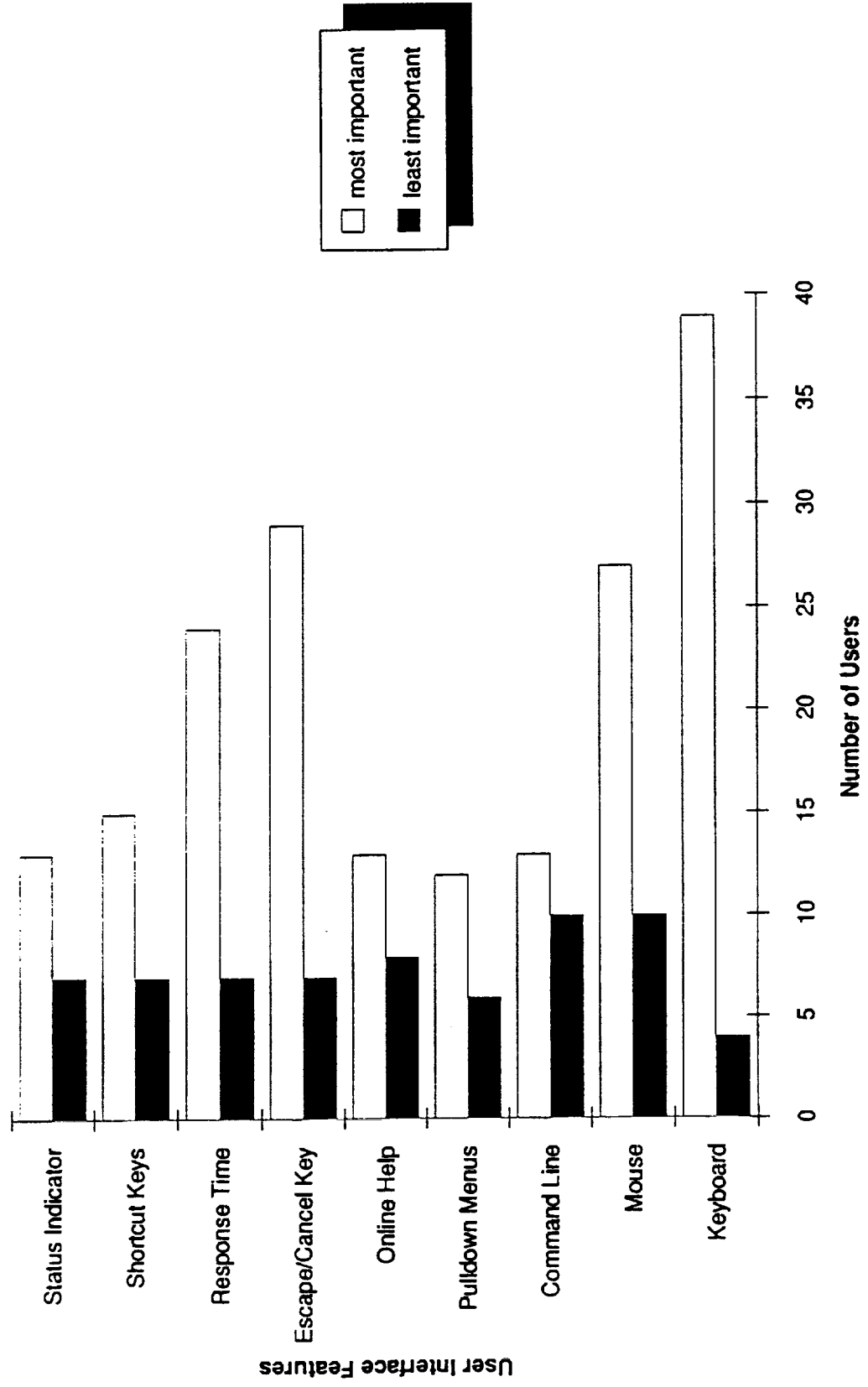


Figure 9 - User Interface Issues

Frequency that Users Reuse a Software Component

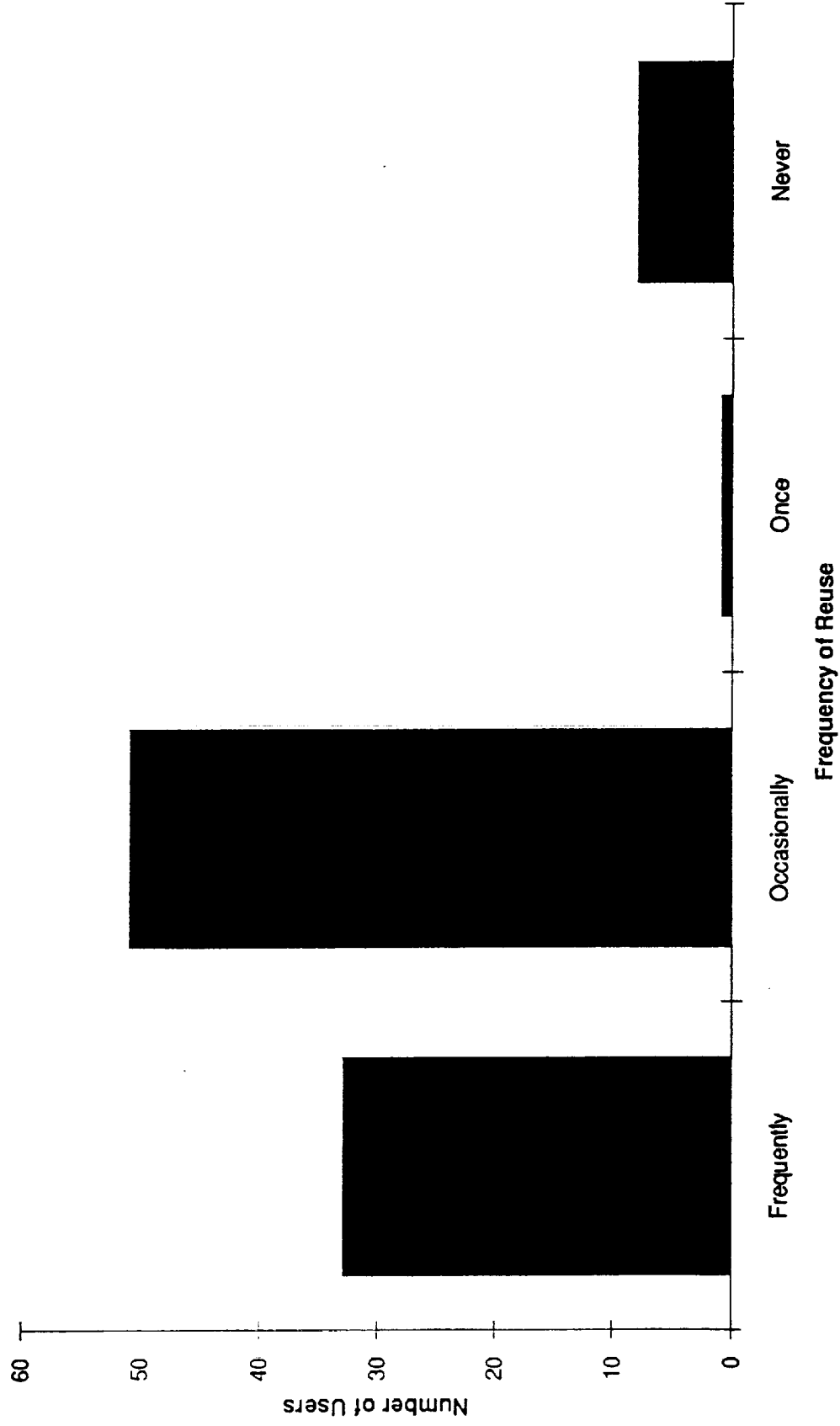


Figure 10 - Software Reuse Issues

System Access for Reusable Software Components

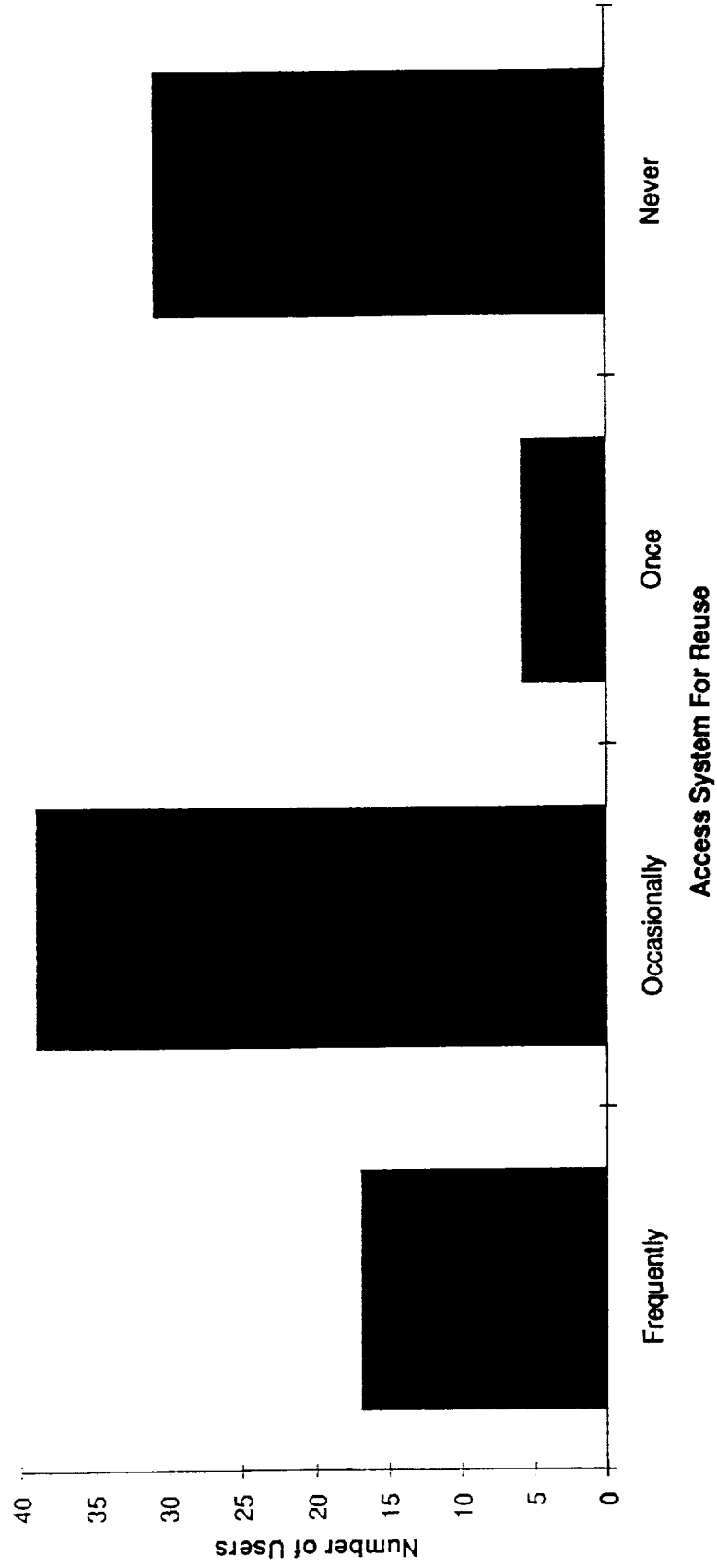


Figure 11 - Software Reuse Issues

System Access to Purchase Software or Library Components

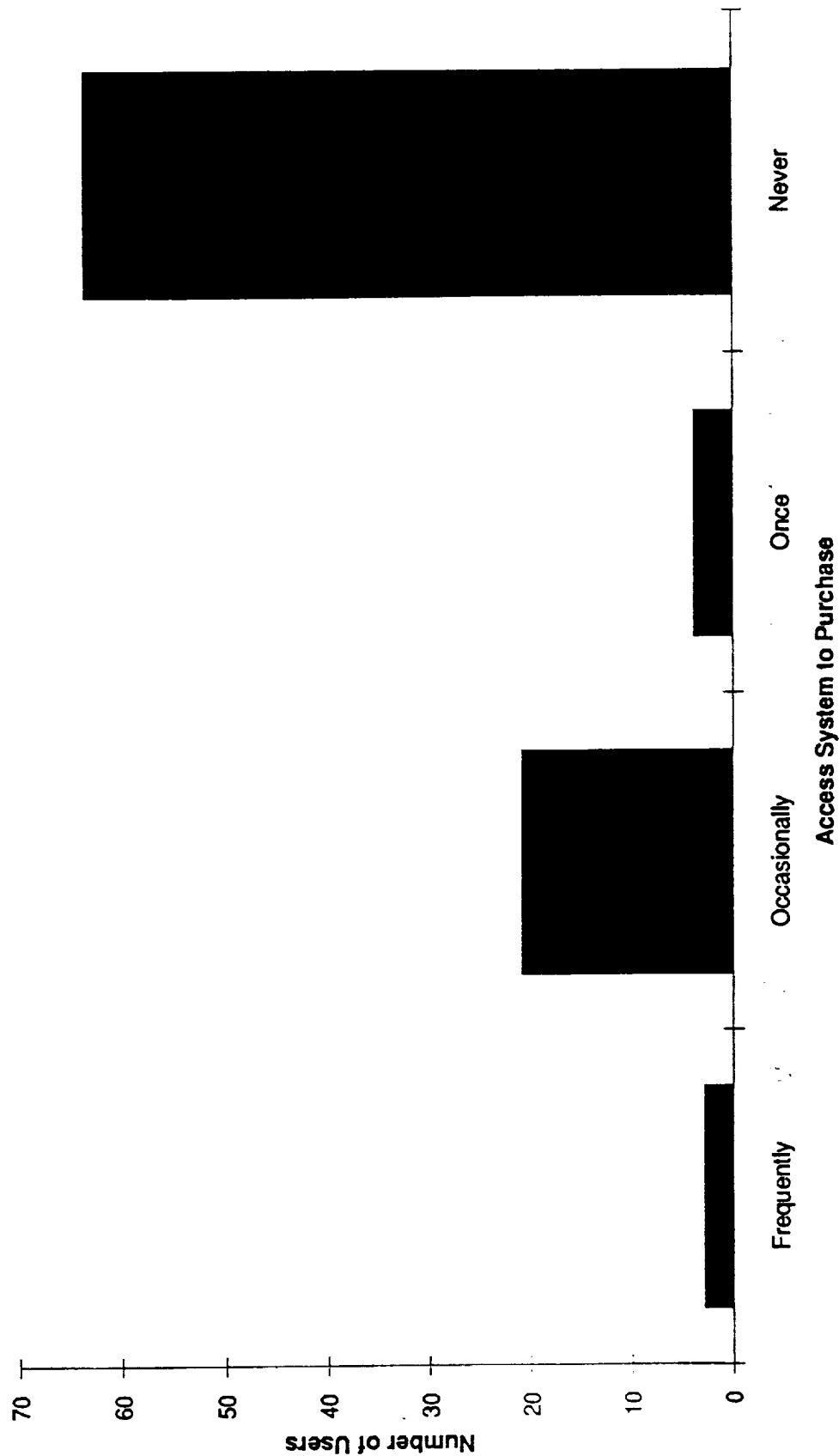


Figure 12 - Software Reuse Issues

Frequency that Users Purchase Software Source Code

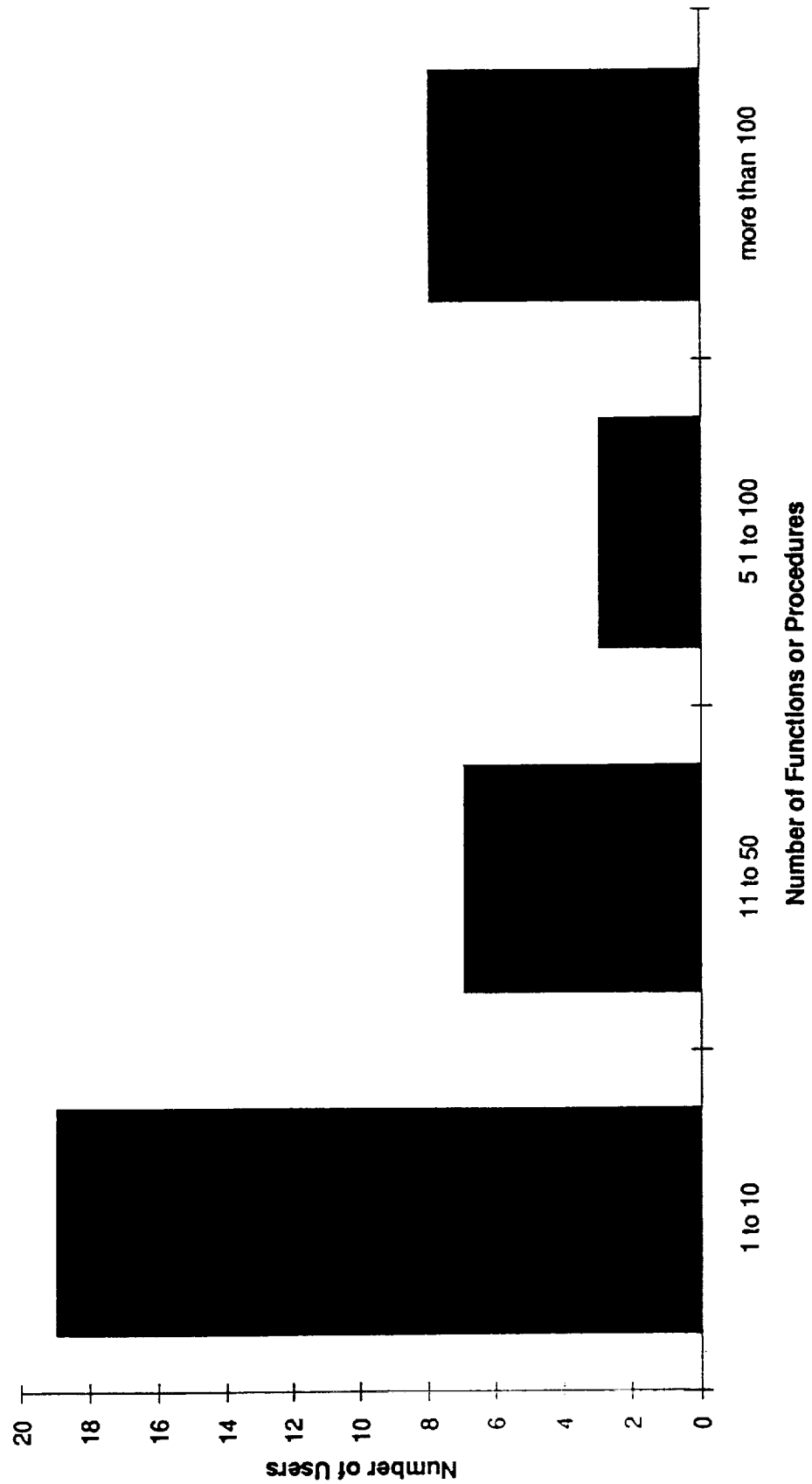


Figure 13 - Software Reuse Issues

Users Rate Search Mechanisms

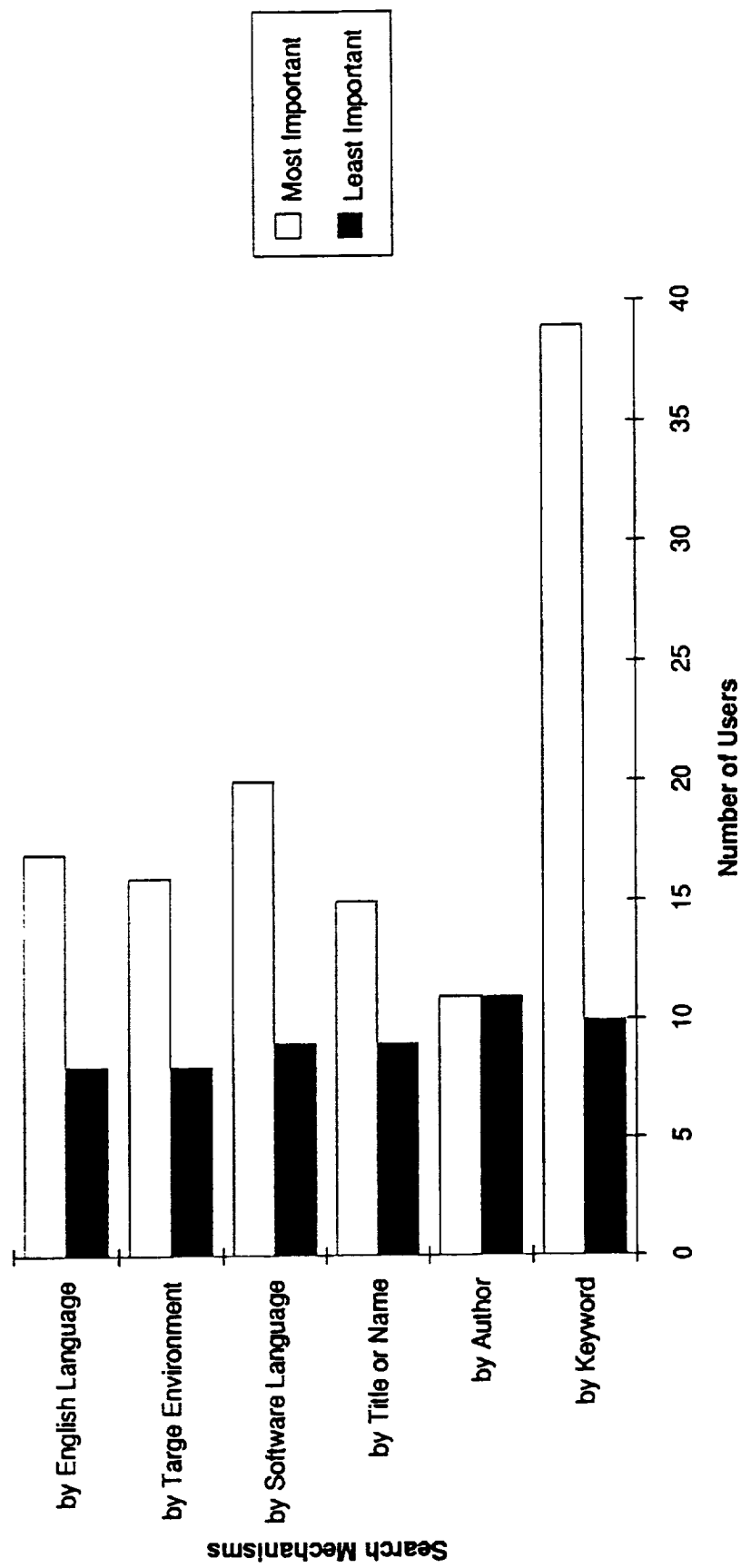


Figure 14 - Search Issues

Types of Services that Users Find Most Helpful

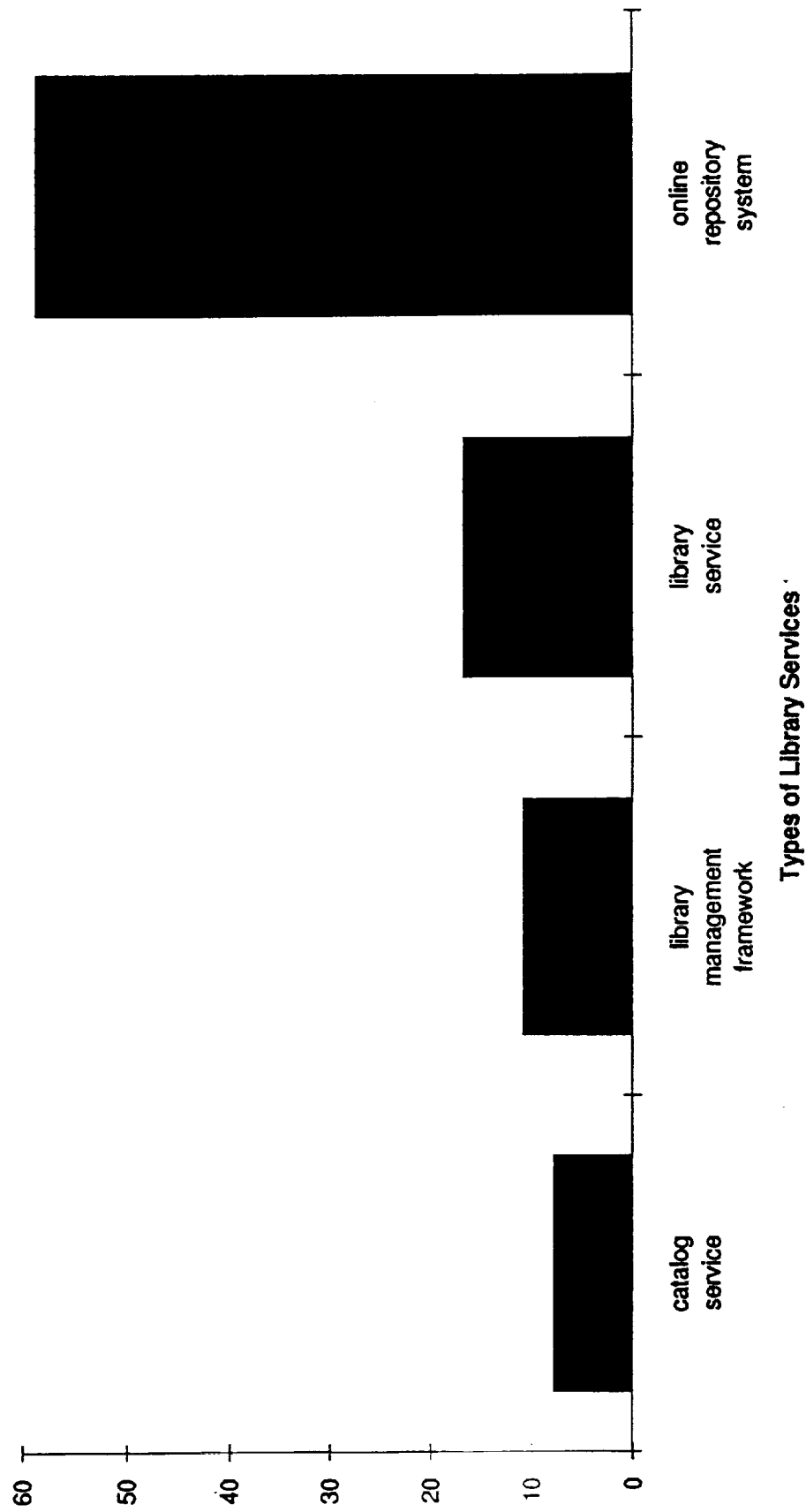


Figure 15 - Search Issues

File Transfer Protocol Preferences

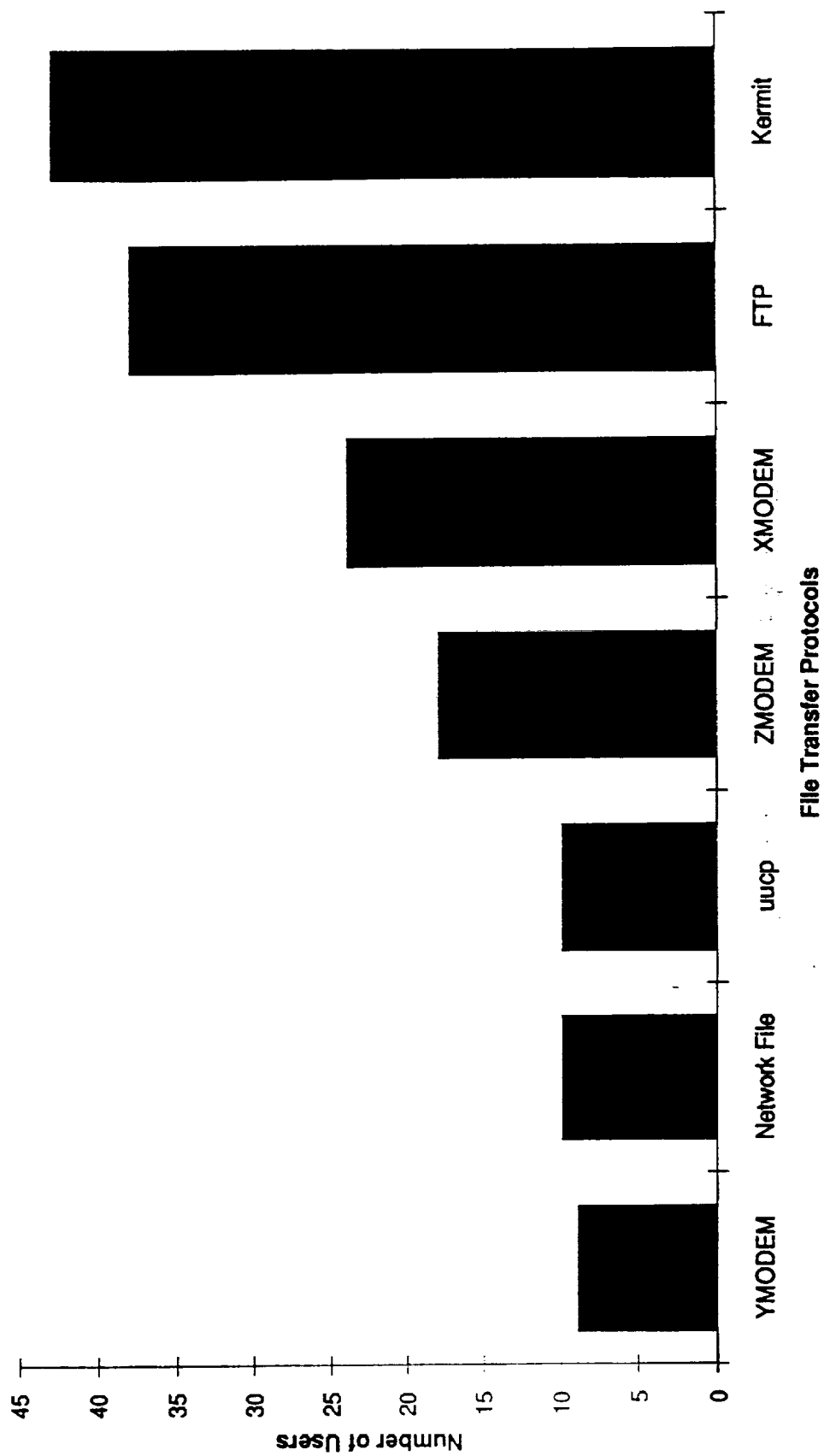


Figure 16 - User Issues

User Affiliations

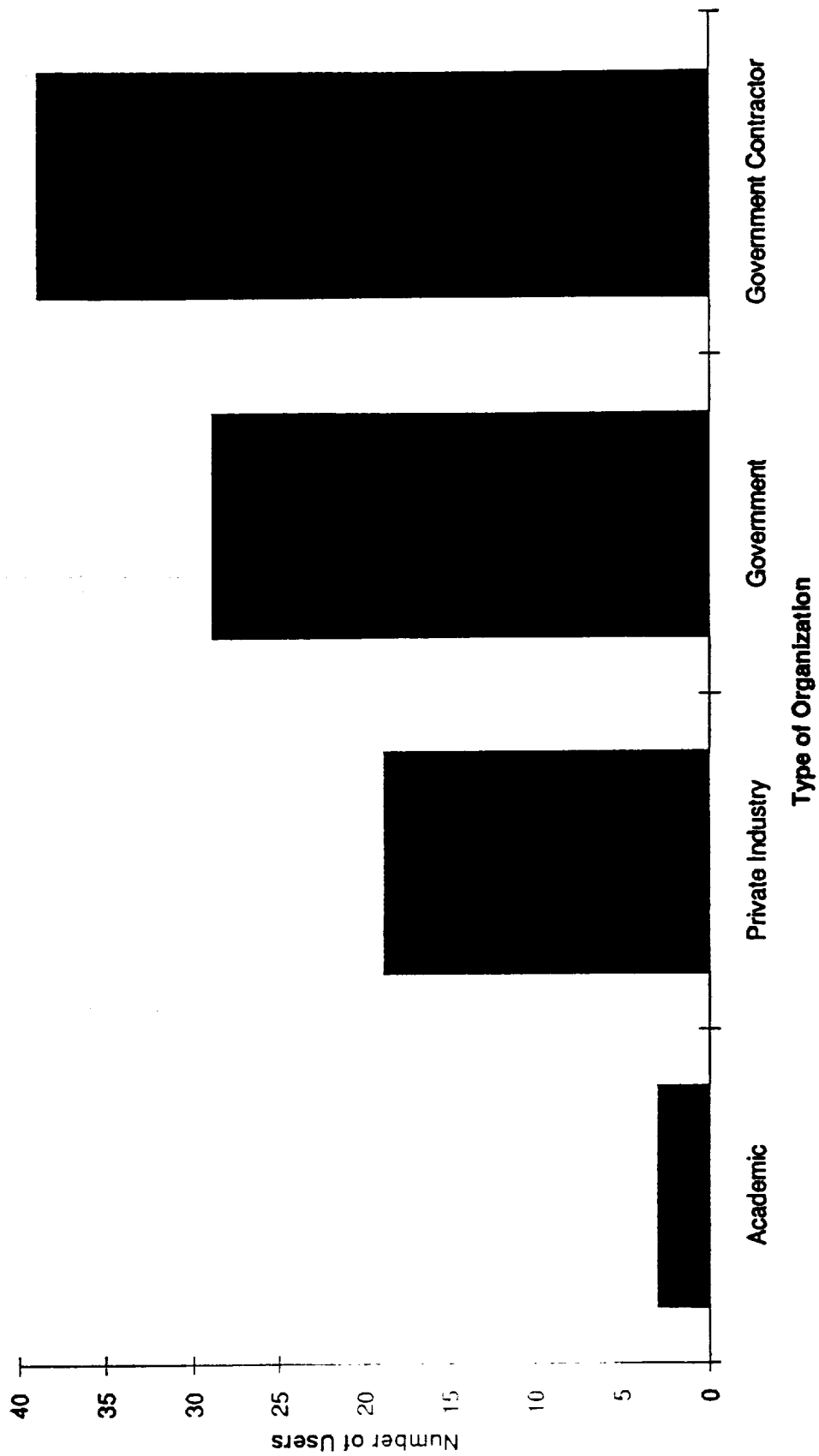


Figure 17 - Demographic Issues

APPENDIX F
DELIVERABLES COVER SHEET

RB.05 GHG Corporation
January 1, 1992

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW #7

Title of Task: GHG's Extensions to NELS 1-1

Subcontractor: GHG Corporation

Cooperative Agreement No. NCC-9-16

Principal Investigator: E.T. DICKERSON

NASA Technical Monitor: E. FRIDGE

Type of Report: _____

Period Covered by Report _____

Due Date: 03/24/92

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

5

GHG Corporation
1300 Hercules, Suite 111
Houston, Texas, 77058

Date: 4 March 1992

Mr. Mark E. Rorvig
Software Technology Branch
Johnson Space Center
NASA

Dear Sir,

Please find, enclosed, two copies of the Design Document, ASV3
Extensions/Functions for NELS 1.1.

Should you have any questions or if I can be of further service,
please don't hesitate to call at (713) 488-8806. Thank you for
your time and effort.

Sincerely,

Robert C. Frederiksen

Robert C. Frederiksen
Configuration/Data Management

cc:

- 1) Mr. E.T. Dickerson
Level 3 RBSE Program Manager
Research Institute for Computing and Information
Systems (RICIS)
University of Houston - Clear Lake,
Houston, Texas
- 2) Dr. Charles McKay
Chief Scientist, RBSE Program
Research Institute for Computing and Information
Systems (RICIS)
University of Houston - Clear Lake,
Houston, Texas
- 3) Ms. Karen Fleming
AdaNET operations Manager
MountainNet
Delslow, West Virginia

IDENTIFICATION AND RELEASE DATA RECORD

DOCUMENT / SPECIFICATION NUMBER: SSDD-00005

NOMENCLATURE OR DOCUMENT TITLE: Design Document For ASV3
Extensions / Functions for
NELS 1.1

PART NUMBER:

VERSION:

RELEASE NUMBER: 0000000000000005

RELEASE DATE 03/04/92

CONTRACT NUMBER: RBSE-FD-R&T-006-0
Subcontract No. 044
Project No. RICIS No. SE. 18,
Cooperative Agreement NCC9-16

PROGRAM TITLE: AdaNET Repository Based Software Engineering
(RBSE)

VENDOR/SUBCONTRACTOR 2Y794 GHG Corporation
CAGE CODE & ADDRESS: 1300 Hercules, Ste 111
Houston, Texas, 77058

ITEM TYPE: H/W ___ DWG. TYPE: _____ DWG. SIZE: _____
SHEET COUNT: _____ DASH NO.: _____
S/W ___ CALL NAME: _____ VERSION: _____

DESCRIPTION (17 LINES MAX): Document created by Dave Henning as
the result of a Memorandum of Understanding between GHG and NASA.

USED ON (EFFECTIVITY): AdaNET NEXT HIGHER ASSEMBLY: N/A

DEVELOPER: David Henning

QA REP: _____

CCB CHAIR: R. C. Frederiksen

DATE: 03/04/92

CCB (RELEASE) CLERK: R. C. Frederiksen DATE: 03/04/92

=====

Design Document
ASV3 Extensions/Functions for NELS 1.1

1.0 Background.

This document describes the GHG Functions and Extensions to be added to the NELS 1.1 product. These functions will implement the "Output Request" capability within the Object Browser. The functions will be implemented in two parts. The first part is code to be added to the Object Browser (X Version) to implement menus allowing the user to request that objects be copied to specific media, or that objects be downloaded to the user's system following a specific protocol, or that the object be printed to one of printers attached to the host system. The second part is shell scripts which support the various menu selections. Additional sScripts to support functions within the GHG Shell (X Version) will also be created along with the X Version of the GHG Shell as initial capability for the March 27 prototype. The scripts will be composed of C shell routines that will accept parameters (primarily file pathways). Certain limitations in functionality will be imposed for the March increment. For instance, the E-Mail functions will invoke Mail instead of Oracle Mail since that has yet to be delivered and the NELS invocation will default to the X-Window version instead of the ASCII version.

2.0 Overview.

2.1 GHG C Source Code.

GHG will prepare source code for delivery to NASA which will implement the "Output Request" function within the NELS Object Browser. It will be up to NASA to oversee the integration of this code into other NELS Development efforts. The GHG source code will create a button on the NELS Object Browser labeled "Output Request". Various references to metadata in the following functions should be interpreted as a linear list in the form:

data label: data value

.

data label: data value

as defined by the class definition of each object.

Design Document
ASV3 Extensions/Functions for NELs 1.1

2.1.1 Activating that button will result in a pulldown menu or an error message window. If no objects had been selected prior to the "Output Request" button activation then the error message window will display the message "You must have selected an object or objects prior to requesting output." If the menu appears, it will list the following selections:

- Copy
- Print
- Download
- Local Copy

2.1.2 The Copy Function.

The Copy Function is intended to allow the user to request that specific objects and their accompanying metadata be transferred to specific media. Selecting the Copy option from the Output Request Menu results in an additional pulldown menu detailing the various devices or media that may be copied to. See Figure 3-1. The user selects an appropriate device causing the following actions to occur:

a. The userid of the user and the current datetime stamp are used for a file name (e.g. userid.datestamp) which is created under the directory \$ASV3/Customer.Service/device where device is a clear text name similar to those listed in Figure 3-1.

b. The following data is written to that file:

```
.device
device code
.enddevice
.metadata
metadata text for object 1 for as many lines as it takes
.endmetadata
.pathway
pathway for object 1
.endpath
.
.
.
.metadata
metadata text for object n as many lines as it takes
.endmetadata
.pathway
pathway for object n
.endpath
```

Design Document
ASV3 Extensions/Functions for NEL5 1.1

c. As each object is written to the file, a record of its access, datetime of access request, and userid of who accessed the file is written to the history file (an Oracle Table).

The Copy Function is then complete. A supporting Client Services function accesses the files produced and completes the requested transfer of data using GHG provided shell scripts.

Device code and clear text name table:

1	3.5 High Density IBM or PC
2	3.5 Low Density IBM or PC
3	3.5 MAC
4	5.25 High Density IBM or PC
5	5.25 Low Density IBM or PC
6	9 Track 1600 BPI tape (TAR)
7	9 Track 1600 BPI tape (ASCII)
8	1/4 inch Cartridge Tape
9	Hardcopy (14.5 by 11)
10	Hardcopy (8.5 by 11)

Figure 3-1

March 4, 1992

SSDD-00005

Design Document
ASV3 Extensions/Functions for NELs 1.1

2.1.3 Print.

Activation of the Print option will cause the display of a pulldown menu listing the available printers. The user will select a printer causing the code to generate a file called \$ASV3/print containing the following data:

```
.device
device code (site dependent, stored in PRINTCAP)
.enddevice
.metadata
metadata text for object 1 for as many lines as it takes
.endmetadata
.pathway
pathway for object 1
.endpath
.metadata
metadata text for object 2 as many lines as it takes
.endmetadata
.pathway
pathway for object 2
.endpath
.
.
.
.metadata
metadata text for object n as many lines as it takes
.endmetadata
.pathway
pathway for object n
.endpath
```

2.1.4 Download.

Activation of the Download option of the Output Request Menu generates a second pulldown menu with the following options:

```
FTP
KERMIT
X-MODEM
Y-MODEM
Z-MODEM
UUCP
```

Initially, the only active options will be FTP and UUCP. This is due to the inability to "escape" back to one's home site and activate a similar utility. The ASCII interface will support all of the above options except FTP. Non-available options will be disabled or "grayed out".

Design Document
ASV3 Extensions/Functions for NELS 1.1

2.1.4.1 FTP.

Selection of the FTP option will create a new window with the data fields "Target Address", "Target Userid", and "Target Password". The window will also have buttons for "OK", "Cancel", and "Help". The user must fill in the fields and press "OK", cancel the operation, or press "help" for an explanation. Pressing "OK" will cause a check for empty fields and an appropriate error message. Assuming that the fields are filled in, the function will build temporary files for metadata of the objects selected and the associated pathways of the objects. If the "script file" already exists (explained in a moment), then data is appended. After recording all of the data, it will ask the user if transfer should be started now or postponed. If the user answers "Now" then it will then activate FTP and pass the pathway of the metadata and object pointers for transfer to the target address using the target userid and password. At the end of the FTP function, the temporary files are deleted. If the user chooses to postpone transfer, then the file is left open so that additional data may be appended later. The naming convention of the temporary files for metadata is \$ASV3/object_file_name.metadata. The "script file" will be named \$ASV3/userid.FTP. The format of the "script file" is:

```
pathway of object 1 metadata
pathway of object 1
.
.
.
pathway of object n metadata
pathway of object n
```

2.1.4.2 Kermit.

Kermit will use much the same procedures as described for FTP. The difference is that the user will be able to set a variety of variables used by Kermit to regulate the transfer protocols. The variables and the format of the ASCII sequence to get their values will be addressed at a later date.

2.1.4.3 X-Modem, Y-Modem, Z-Modem.

These protocols are closely related. Setting some values controlling Xmodem causes the use of Y-Modem or Z-Modem. The objects and metadata will use much the same procedures as described for FTP. The difference is that the user will be able to set a variety of variables used by Xmodem to regulate the transfer protocols. The variables and the format of the ASCII sequence to get their values will be addressed at a later date.

March 4, 1992

SSDD-00005

Design Document
ASV3 Extensions/Functions for NELS 1.1

2.1.4.4 UUCP.

UUCP is a form of Internet Mail. Selection of this option will display a request for the target or "bang" address. UUCP will use much the same procedures for objects and associated metadata files as described for FTP.

2.1.5 Local Copy.

The local copy is provided as a debug tool as much as anything else. It is only available to librarians and not average users. It will create a file named \$HOME/lcopy.datetime with the following data:

```
.metadata
metadata text for object 1 for as many lines as it takes
.endmetadata
.pathway
pathway for object 1
.endpath
.metadata
metadata text for object 2 as many lines as it takes
.endmetadata
.pathway
pathway for object 2
.endpath
.
.
.
.metadata
metadata text for object n as many lines as it takes
.endmetadata
.pathway
pathway for object n
.endpath
```

A shell script file will then be called to create metadata files and copies of the associated objects in the user's home directory. The same routines which create metadata files used for FTP (and other) transfers will be used following the same file naming conventions. The shell script will not delete the resulting files (unlike the transfer functions).

2.1.6 Exit Processing.

GHG will provide code to be inserted into the exit processing functions of NELS 1.1 which will query the status of a global variable to determine whether outstanding download requests are pending. If download requests are pending the user will be asked to either start the transmission or cancel the request.

Design Document
ASV3 Extensions/Functions for NELS 1.1

2.2 GHG Shell Scripts.

Scripts will be created to support the following functions:

<u>Script Name</u>	<u>Description</u>
Local_Copy	Local copy to home directory
Local_Print	Local print to site printer
Change_Password	Change the user password
Set_Term	Set the term type
Lan_Copy	Modified version of Local_Copy
Tar_Tape	Transfer a list of files to a 1600 BPI Tar Tape
Ascii_Tape	Transfer a list of files to a 1600 BPI Ascii Tape
Cart_Tape	Transfer a list of files to a 1/4 inch Cartridge Tape
Wide_Print	Print a list of files to 14.5x11 forms
Side_Print	Print a list of files to 8.5x11 forms
Print_Stat	Dump NELS statistics using predefined Oracle report

The majority of the scripts are very simple. The only complication is providing the capability for multiple selections. Multiple selections will be handled in a uniform way. Multiple selections will be accumulated by appending metadata directive (.metadata) followed by the metadata for the object, followed by the pathway directive (.pathway) and the full file pathway to an ascii file in the users home directory. NELS will open this file for append every time the user indicates that a file should be marked for download, copy or print. There will be a separate file for download, copy and print. The GHG scripts will use this file as input (required parameter is filename) and destroy the file upon completion of the function. The media request is a little more complicated because more data is required than just file pathways. The media request will use a set of pseudo directives to separate different types of data needed to detail the files to be transferred and the mailing information.

This approach will provide for cumulative identification of objects as well as allowing the list to be used as a clear text list of instructions if customer service chooses to do the job manually. The File name for this media directive should be userid.datestamp which will allow for storage of multiple requests by the same user if necessary. The ".end" directives are required for any multi-line data values.

March 4, 1992

SSDD-00005

Design Document
ASV3 Extensions/Functions for NELs 1.1

The customer service personnel can access these media request files and either use download software from the GHG shell or UNIX utilities to print, TAR, or copy the files to the appropriate devices. The GHG shell will not handle downloads of offline files or files stored on other hosts. Nor will the download handle dumps to tape or print with specific forms. Inclusion of the device code allows for exception handling to detect these types of problems.

3.0 Detailed Interface Description.

All script names are case sensitive.

3.1. Local Copy.

3.1.1 Description.

This routine will copy all files listed within the input file to the user's HOME directory. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be copied because of invalid pathways or lack of permissions.

Script name: Local_Code

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

3.2 Local Print.

3.2.1 Description.

This routine will print all files listed within the input file to the printer requested. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be printed because of invalid pathways, lack of permissions, or bad format.

Script name: Local_Print

Required parameters:

character string, full pathway of file list file.

character string, printcap code for required printer.

Return code: 0 for success, greater than 0 is error.

Design Document
ASV3 Extensions/Functions for NELS 1.1

3.3 Change the users password.

3.3.1 Description.

This routine will allow the user to specify a new password. The routine will return a code of 0 for successful, 1 unsuccessful.

Script name: Change_Password

Required parameters:

none.

Return code: 0 for success, greater than 0 is error.

3.4 Set Terminal Type.

3.4.1 Description.

This routine will allow the user to specify a different terminal type. The routine will return a code of 0 for successful, 1 unsuccessful.

Script name: Set_Term

Required parameters:

character string, entry code for termcap.

Return code: 0 for success, greater than 0 is error.

3.5 Copy Requested Objects to Media.

3.5.1 Description.

There are really two classes of scripts here. Most of the PC/MAC compatible media is not a device directly connected to the host. The necessitates Customer Service transferring these files to a PC or MAC first. This may be done by executing a variant of the local_copy script against the files stored in the various "device" directories followed by FTP or Kermit as an interactive user. For those devices directly connected to the host, specific scripts can be executed using the files stored in the appropriate directory. This approach allows customer service to both control the sequence of media copy and the priority. The file naming conventions userid.datetime allows customer service determine how old the outstanding requests are by simply executing a directory command from within UNIX. The routines will return a code of 0 for successful, 1 for missing parameters, 2 bad device code, 3 for partial success.

Script name: Lan_Copy

Function: Creates a copy of requested files in the home directory of the Customer Service representative.

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

Design Document
ASV3 Extensions/Functions for NELS 1.1

Script name: Tar_Tape
Function: Creates a copy of requested files on a 1600 BPI tape in TAR format.
Required parameters:
character string, full pathway of file list file.
Return code: 0 for success, greater than 0 is error.

Script name: Ascii_Tape
Function: Creates a copy of requested files on a 1600 BPI tape in ASCII format.
Required parameters:
character string, full pathway of file list file.
Return code: 0 for success, greater than 0 is error.

Script name: Cart_Tape
Function: Creates a copy of requested files on a 1/4 inch Cartridge Tape.
Required parameters:
character string, full pathway of file list file.
Return code: 0 for success, greater than 0 is error.

Script name: Wide_Print
Function: Prints the associated metadata and objects on 14.5 x 11 form.
Required parameters:
character string, full pathway of file list file.
Return code: 0 for success, greater than 0 is error.

Script name: Side_Print
Function: Prints the associated metadata and objects on 8.5 x 11 form (landscape mode).
Required parameters:
character string, full pathway of file list file.
Return code: 0 for success, greater than 0 is error.

3.6 Print Current Statistics Report.

3.6.1 Description.

This routine will allow the user to request a predefined report. The routine will return a code of 0 for successful, 1 for missing parameters, 2 bad printer code, 3 for bad report name.

Script name: Print_Stat
Required parameters:
character string, Oracle name for the report to be printed.
character string, printcap code for required printer.
Return code: 0 for success, greater than 0 is error.

March 4, 1992

SSDD-00005

Design Document
ASV3 Extensions/Functions for NELS 1.1

4.0 Schedule For Development.

Delivery of Initial NELS 1.1 code to GHG from NASA	28 Feb 1992
Development of GHG code and scripts for GHG Extensions to NELS 1.1	4 Mar 1992 - 16 Mar 1992
Unit Testing of GHG code	16 Mar 1992 - 24 Mar 1992
Test Plan Development for GHG Code and Scripts	4 Mar 1992 - 16 Mar 1992
Submission of Code, Scripts and proposed Test Plan for GHG Extensions and Scripts	24 Mar 1992



GHG Corporation

The Decisive Edge in Software Engineering

FACSIMILE TRANSMISSION

Number of Sheets: 14
(This one included)

TO: DR. C.W. McKay
Individual

RICIS / U. of H.
Organization

Address

Address

Telephone

Facsimile

FROM:

BOB FREDERIKSEN
C/DM

GHG CORPORATION
1300 Hercules, Suite 111
Houston, Texas 77058
Telephone: (713) 488-8806
Faxphone: (713) 488-1838

Comments:

COPY FOLLOWS BY MAIL



GHG Corporation

The Decisive Edge in Software Engineering

FACSIMILE TRANSMISSION

Number of Sheets: 14
(This one included)

TO: E.T. DICKERSON
Individual
RICIS / U. of H.
Organization

Address

Address

Telephone

Facsimile

FROM:

BOB FREDERIKSEN
C/D M

GHG CORPORATION
1300 Hercules, Suite 111
Houston, Texas 77058
Telephone: (713) 488-8806
Faxphone: (713) 488-1838

=====
Comments:

COPY WILL FOLLOW BY MAIL

APPENDIX F
DELIVERABLES COVER SHEET

RB.05 GHG Corporation
January 1, 1992

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW A 8

Title of Task: Problem/change Request Report - NELS CR #254

Subcontractor: GHG Corporation

Cooperative Agreement No. NCC-9-16

Principal Investigator: E.T. DICKERSON

NASA Technical Monitor: E. FRIDGE

Type of Report: _____

Period Covered by Report _____

Due Date: 03/24/92

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

PROBLEM / CHANGE REQUEST FORM

PROBLEM REPORT TYPE: DR__ CR XX AR__ RID__

PROBLEM REPORT NUMBER: NASA NELS 1.1 CR#354 CAN: 000000000001

PROBLEM REPORT SHORT TITLE: Output Capability for Browser Screen

DATE DISCOVERED: 01/10/92 PRIORITY: 01

DESCRIPTION: Provide output capability from the Browser Screen and implement into NELS, Revision 1.1.

ORIGINATOR: Karen Fleming

DATE: 01/10/92

ORIGINATOR AGENCY: MountainNet

PHONE:

CORRECTIVE ACTION SUMMARY:

Modified Barrios software, Part Number GHG.BETA1_RELEASE.031392.tar.z to include print capability from the Browser Screen.

PROGRAM MANAGER: Jeff McGee

DATE SUBMITTED: 03/24/92

RETEST RESULTS:

RETESTED BY:

DATE:

PROBLEM REPORT

DISPOSITION: PASS__ FAIL__ WITHDRAWN__ HOLD__

COMMENTS:

PROBLEM ANALYSIS FORM

=====

PROBLEM REPORT NUMBER: NASA NELS 1.1 CR#354 CAN: 000000000001

=====

PROBLEM ANALYSIS:

See Attachment 1 to this Change Package.

=====

PROPOSED SOLUTION:

UNITS AFFECTED: (Include Revision Level and OLD/NEW Software Versions)

SOFTWARE: Module Id.	Versions				
	Old	New			
browser.c	1.4	1.5	help.c	1.4	1.6
oms.c	1.3	1.5	helpindex	1.1	1.2
request.output	--	1.1	ghg.h	--	1.1
pipe.h	--	1.1	cart.l	--	1.1
cart.y	--	1.1	cartmain.c	--	1.1
copy.l	--	1.1	copy.y	--	1.1
ftp.l	--	1.1	ftp.y	--	1.1
ftpmain.c	--	1.1	main.c	--	1.1
media.l	--	1.1	media.y	--	1.1
print.l	--	1.1	print.y	--	1.1
uucp.l	--	1.1	uucp.y	--	1.1
request.output.copy	--	1.1			
request.output.download	--	1.1			
request.output.ftp	--	1.1			
request.output.local_copy	--	1.1			
request.output.now_or_later	--	1.1			
request.output.print	--	1.1			
request.output.session	--	1.1			
request.output.uucp	--	1.1			

DOCUMENTATION: autoLib 4.0 SYSTEM TEST PLAN, August 1991

=====

ALTERNATE SOLUTIONS:

RECOMMENDED COURSE OF ACTION:

ANALYST: Dave Henning, GHG

DATE:

=====

ANALYSIS REVIEW

APPROVED XX

HOLD ____

REJECTED ____

PROGRAM MANAGER Jeff Mcgee

DATE:

=====

PROBLEM / CHANGE REQUEST

PROBLEM REPORT TYPE: DR__ CR XX AR__ RID__

PROBLEM REPORT NUMBER: NASA NELS 1.1 CR#355 CAN: 000000000002

PROBLEM REPORT SHORT TITLE: Change Copy Script

DATE DISCOVERED: 03/07/92 PRIORITY: 01

DESCRIPTION: Change copy script to list possible selections;
action should be similar to print.

ORIGINATOR: Karen Flemming

DATE: 03/07/92

ORIGINATOR AGENCY: MountainNET

PHONE:

CORRECTIVE ACTION SUMMARY:

Modified Barrios software, Part Number GHG.BETA1_RELEASE.031392.tar.z
to list possible selections.

PROGRAM MANAGER: Jeff McGee

DATE SUBMITTED:

RETEST RESULTS:

RETESTED BY:

DATE:

PROBLEM REPORT

DISPOSITION: PASS __ FAIL __ WITHDRAWN __ HOLD __

COMMENTS:

ORIGINAL PAGE IS
OF POOR QUALITY

PROBLEM ANALYSIS FORM

=====

PROBLEM REPORT NUMBER: NASA NELS CR # 355 CAN: 0000000000002

PROBLEM ANALYSIS:

See Attachment 1 to the Change Package.

=====

PROPOSED SOLUTION:

UNITS AFFECTED: (Include Revision Level and OLN/NEW Version Levels)

SOFTWARE: See list for CR 354.

DOCUMENTATION:

HARDWARE:

DRAWINGS:

ESTIMATED REPAIR TIME IN MANHOURS: (Includes Internal Test)

=====

ALTERNATE SOLUTIONS:

RECOMMENDED COURSE OF ACTION:

ANALYST:

DATE:

=====

ANALYSIS REVIEW

APPROVED XX

HOLD ____

REJECTED ____

PROGRAM MANAGER Jeff McGee

DATE:

Statement of Capabilities:

This software release conforms to the Interface Control Document, ASV3 Extensions/Functions for NEL3 1.1, SSDD-00005, 4 Mar 1992 and the memo, Proposed File Naming Conventions used within the GHG Extensions, 13 Mar 1992 with the following exceptions:

1. Scripts are not provided to support a request for copy to 9 Track 1600 BPI tape in either the TAR or ASCII format.
2. Scripts are not provided to support a request for copy to 14.5x11 hardcopy or 8.5x11 hardcopy.
3. Scripts implementing media copy to IBM or MAC media consist of FTPing files to a PC or MAC in FTP Server mode. Once the files arrive at the PC/Mac they can be transferred to whatever media has been requested as a manual offline process.
4. All references to pathways are implemented relative to \$AUTOLIB_HOME instead of \$ASV3.

All exceptions reflect the lack of appropriate testbed equipment to verify capabilities.

All software has been developed and tested on UHCL computers. The testbed software used BTI code from 18 Mar 1992. The following functions have been verified:

1. The COPY function (Request for Media) has been executed for TAR format 1/4 inch Cartridge tapes and IBM diskettes. GHG did not have access to a MAC with a FTP Server to verify transmission from UHCL to a MAC.
2. The PRINT function has been verified by printing text and Postscript documents on GHG printers from the UHCL computer.
3. The Local Copy has been fully tested.
4. The Download function supports FTP and UUCP only. The target environment was X users who wouldn't be likely to download via Kermit or Zmodem.
5. All help buttons and dialogs which are part of the GHG extensions have been validated as working.
6. The Test Plan update has been fully executed and verified.

The following files reflect either new GHG provided source or GHG updated source. BTI will have to integrate these files into the NELS 1.1 baseline.

<u>C Source</u>	<u>Function</u>
ghg.h	Include for GHG Extensions
ghg.c	Source for X interface GHG Extensions
pipe.h	Include for interface to FTP
pipe.c	Source for interface to FTP
makefile	Updated BTI make file
browser.c	Updated BTI source
help.c	Updated BTI source
oms.c	Updated BTI source
<u>Scripts</u>	<u>Function</u>
media.y	Yacc source for creation of media queues
media.l	Lex source for creation of media queues
copy.y	Yacc source for local copy
copy.l	Lex source for local copy
print.y	Yacc source for print
print.l	Lex source for print
main.c	Common function interface for media/local copy/print
ftp.y	Yacc source for FTP to FTP Server
ftp.l	Lex source for FTP to FTP Server
ftpmain.c	Entry or FTP to FTP Server
ftp.h	Include for FTP to FTP Server
cart.y	Yacc Source for TAR to Cartridge
cart.l	Lex source for TAR to Cartridge
cart.c	Entry for TAR to Cartridge
makefile	Make file for scripts
<u>Help Files</u>	<u>Function</u>
helpindex	Updated BTI file
request.output	main help for output request button
request.output.copy	help for copy function
request.output.download	help for download functions
request.output.ftp	help for ftp form
request.output.local_copy	help for local copy function
request.output.now_or_later	help for session capability
request.output.print	help for print functions
request.output.session	help for exit functions
request.output.uucp	help for uucp form

TABLE OF CONTENTS

SECTION 1.....	3
INTRODUCTION.....	3
SECTION 2.....	4
TEST OBJECTIVES.....	4
SECTION 3.....	5
SRS / TEST PROCEDURE CROSS-REFERENCE MATRIX.....	5
SECTION 4.....	13
TEST IMPLEMENTATION.....	13
4.1 HARDWARE AND SOFTWARE REQUIREMENTS.....	13
4.2 LOCATION AND SCHEDULE.....	13
4.3 PREPARATION OF INPUTS.....	13
SECTION 5.....	14
TEST CASES.....	14
5.1 USER INTERFACE REQUIREMENTS.....	14
5.1.1 Help.....	14
5.1.2 Screen Layout.....	30
5.1.4 Mouse Interface.....	46
5.1.5 Menu Driven Interface.....	47
5.1.6 System Exit.....	47
5.2 USER FUNCTIONS.....	61
5.2.1 Browsing.....	61
5.2.2 Retrieval.....	79
5.2.3 Retrieval Options.....	99
5.2.4 Tools.....	101
5.2.5 Reporting.....	105
5.2.6 User Profile.....	107
5.3 LIBRARIAN FUNCTIONS.....	111
5.3.1 Access Privileges.....	111
5.3.2 Object Classes.....	117
5.3.3 Collections.....	125
5.3.4 Tools.....	130
5.3.5 Objects.....	133
5.3.6 Synonym Table.....	137
5.4 AUDIT.....	138
5.5 OUTPUT REQUEST.....	139

5.5 Output Request

55-001) Test whether the user can Output Request from the Object Browser.

Action: (From the Main Menu or Window)

Click on BROWSE.

This should present a submenu containing Object Classes and Objects.

Click on OBJECTS.

This should present the Object Browser window containing a menu of objects associated with the current or subordinate collection depending on the current active search strategy. If no objects are found the window will be blank. Use the VIEW button and associated options on the main window to change collections. Repeat the above procedures until a list of objects is found.

Click on one or more objects. This the method for SELECTING Objects.

Click on the FILE button of the object browser.

This should reveal a menu containing Request Object, Close Window and Exit Program.

Click on the Request Object button.

This should reveal a menu containing Copy, Print, Local Copy, Download, Output Help.

Date:

Result:

55-002) Test whether the user can activate Help from the Output Request menu.

Click on the Output Help button of the Output Request menu.

The help window should appear with help describing each of the Output Request functions.

Close the Help Window by Clicking on the FILE button of the Help window, then click on Close Window.

Date:

Result:

55-003) Test whether the user can request a copy of objects to selected media.

Click on a variety of Objects shown in the Object Browser.

Click on the FILE Button of the Object Browser.

Click on the Output Request Button of the FILE menu.

Click on the Copy button of the Output Request menu.

This should present a question box to allow the user to specify whether the Copy Request should be processed now or later.

Click on the Help Button of the question box.

This should present a help window explaining the concept of session copies versus immediate copies.

Close the Help window by clicking on the FILE button of the help window and selecting Exit.

This will return focus to the question box.

Click on NOW.

This will present a menu window containing various devices that the selected objects can be copied to.

Click on IBM 3.5 HD (High Density).

There will be a short pause and the window will close returning focus to the object browser. The requested objects and associated metadata files will have been copied to a directory entitled IBM3.5HD subordinate to the directory contained in the \$ASV environmental variable. Inspection of that directory should show pairs of files (one pair for each object selected).

Date:

Result:

55-004) Test whether the user can print a copy of objects to a host connected printer.

Click on text or Postscript Objects shown in the Object Browser.

Click on the FILE Button of the Object Browser.

Click on the Output Request Button of the FILE menu.

Click on the Print button of the Output Request menu.

This should present a question box to allow the user to specify whether the Print Request should be processed now or later.

Click on NOW.

This will present a menu window containing various printers that the selected objects can be printed to.

Click on a Postscript print if you have selected Postscript objects or a line printer for text objects.

There will be a short pause and the window will close returning focus to the object browser. The requested objects and associated metadata files will print to the requested printer.

Date:

Result:

55-005) Test whether the user can download a copy of objects to a remotely connected computer.

Click on a variety of objects shown in the Object Browser.

Click on the FILE Button of the Object Browser.

Click on the Output Request Button of the FILE menu.

Click on the Download button of the Output Request menu.

This will present a submenu listing FTP and UUCP as options (other options including Kermit and Zmodem will be delivered in following releases).

Click on FTP.

This should present a question box to allow the user to specify whether the Print Request should be processed now or later.

Click on NOW.

This will present a window where the user should enter the target machine address, target machine signon userid and password.

Click on the Help Button.

This will explain formats required for the FTP fields.

Close the help window.

Enter a valid FTP address.

Enter a valid Userid and Password for that address.

Click on OK.

There will be a short pause and the window will close returning focus to the object browser. The requested objects and associated metadata files will be transmitted to the requested site.

Date:

Result:

55-006) Test whether the user can generate a copy of objects to his home directory.

Click on a variety of objects shown in the Object Browser.

Click on the FILE Button of the Object Browser.

Click on the Output Request Button of the FILE menu.

Click on the Local Copy button of the Output Request menu.

This should present a question box to allow the user to specify whether the Copy should be down now or later.

Click on NOW.

There will be a short pause and the window will close returning focus to the object browser. The requested objects and associated metadata files will print to the requested printer.

Date:

Result:

APPENDIX F
DELIVERABLES COVER SHEET

RB.05 GHG Corporation
January 1, 1992

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW # 9.

Title of Task: Design document for GHG extensions to NELS 1-2

Subcontractor: GHG Corporation

Cooperative Agreement No. NCC-9-16

Principal Investigator: E.T. DICKERSON

NASA Technical Monitor: E. FRIDGE

Type of Report: _____

Period Covered by Report _____

Due Date: 04/01/92

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

9

CONFIGURATION UNIT
IDENTIFICATION AND RELEASE DATA RECORD

DOCUMENT / SPECIFICATION NUMBER: MOU-00002

NOMENCLATURE OR DOCUMENT TITLE: Design Document for GHG
Extensions for NELS 1.2

PART NUMBER: REV. LEVEL: Init.

RELEASE NUMBER: 0000000000000009 RELEASE DATE: 04/01/92

CONTRACT NUMBER: RBSE-FD-R&T-006-0
Subcontract No. 044
Project No. RICIS No. SE. 18,
Cooperative Agreement NCC9-16

PROGRAM TITLE: Repository Based Software Engineering (RBSE)
Program (AdaNET)

VENDOR/SUBCONTRACTOR ZY794 GHG Corporation
CAGE CODE & ADDRESS: 1300 Hercules, Ste 111
Houston, Texas, 77058

ITEM TYPE: H/W ___ DWG. TYPE: _____ DWG. SIZE: _____
SHEET COUNT: _____ DASH NO.: _____
S/W ___ CALL NAME: _____ VERSION: _____

DESCRIPTION (17 LINES MAX): A design document describing both the user interface
and the system interface between NELS and the GHG extensions. MOU generated at
the request of Dr. C.W. McKay (RICIS), U of Houston (also a recipient)

USED ON (EFFECTIVITY): AdaNet NEXT HIGHER ASSEMBLY: N/A

DEVELOPER: D. Henning QA REP:

CCB CHAIR: R. C. Frederiksen DATE: 04/01/92

CCB (RELEASE) CLERK: Frederiksen DATE: 04/01/92

April 1, 1992

To: Dr. Charles McKay, RICIS-UHCL

From: D. Henning, GHG Corporation

Info: E.T. Dickerson, RICIS-UHCL
M. Rorvig, STB, NASA
E. Fridge, NASA
J. McGee, GHG Corporation
K. Fleming, MountainNet
D. Auty, Softech

Subject: Design Document for the GHG Extentions for NELS 1.2

As per Dr. McKay's request, GHG is providing a design document describing both the user interface and the system interface between NELS and the GHG Extentions. The document is attached.

Design Document for the ASCII
GHG Extensions to NELS 1.2

1.0 Background.

This document describes the ASCII version of the GHG Functions and Extensions to be added to the NELS 1.2 product. These functions will implement the "Output Request" capability within the Object Browser. The general look and feel for these functions was derived from the ASCII interface present in NELS 1.0. The functions will be implemented in two parts. The first part is code to be added to the Object Browser to implement menus allowing the user to request that objects be copied to specific media, or that objects be downloaded to the user's system following a specific protocol, or that the object be printed to one of printers attached to the host system. The second part is shell scripts which support the various menu selections.

2.0 Overview.

2.1 GHG C Source Code.

GHG will prepare source code for delivery to NASA which will implement the "Output Request" function within the NELS Object Browser. It will be up to NASA to oversee the integration of this code into other NELS Development efforts. The GHG source code will generate menus to be inserted when the user selects the "Output Request" function. Various references to metadata in the following functions should be interpreted as a linear list in the form:

data label: data value

.

data label: data value

as defined by the class definition of each object.

2.1.1 Typing the code for Output Request will result in a menu or an error message. If no objects had been selected prior to the "Output Request" then the error message, "You must have selected an object or objects prior to requesting output" will display. If the menu appears, it will list the following selections:

```
c [start] [stop] Copy Selected Objects to computer media.
p [start] [stop] Print Selected Objects to host printers.
d [start] [stop] Download Selected Objects to your site.
l [start] [stop] Local Copy (Copy Objects to $HOME directory
                  [Librarians Only]).
h Output Help.
```

Design Document for the ASCII
GHG Extensions to NELS 1.2

The user will be able to specify ranges both contiguous and discontiguous (start - stop or start, next, stop). The parsing routine used will be the same parser used by similar capability commands in other sections of NELS.

2.1.2 The Copy Function.

The Copy Function is intended to allow the user to request that specific objects and their accompanying metadata be transferred to specific media. Selecting the Copy option from the Output Request Menu results in display of the various devices or media that may be copied to. See Figure 3-1. The user selects an appropriate device causing the following actions to occur:

a. The userid of the user and the current datetime stamp are used for a file name (e.g. userid.datestamp) which is created under the directory \$ASV3/Customer.Service/device where device is a clear text name similar to those listed in Figure 3-1.

b. The following data is written to that file:

```
.device
device code
.enddevice
.metadata
metadata text for object 1 for as many lines as it takes
.endmetadata
.pathway
pathway for object 1
.endpath
.
.
.metadata
metadata text for object n as many lines as it takes
.endmetadata
.pathway
pathway for object n
.endpath
```

c. As each object is written to the file, a record of its access, datetime of access request, and userid of who accessed the file is written to the history file (an Oracle Table).

The Copy Function is then complete. A supporting Client Services function accesses the files produced and completes the requested transfer of data using GHG provided shell scripts.

Design Document for the ASCII
GHG Extensions to NELS 1.2

Device code and clear text name table:

	1	3.5 High Density IBM or PC
	2	3.5 Low Density IBM or PC
	3	3.5 MAC
	4	5.25 High Density IBM or PC
	5	5.25 Low Density IBM or PC
*	6	9 Track 1600 BPI tape (TAR)
*	7	9 Track 1600 BPI tape (ASCII)
	8	1/4 inch Cartridge Tape
*	9	Hardcopy (14.5 by 11)
*	10	Hardcopy (8.5 by 11)

* Devices not supported until target machine is
installed at user site.

Figure 3-1

2.1.3 Print.

Activation of the Print option will cause the display of a menu of available printers. The user will select a printer causing the code to generate a file called \$ASV3/print containing the following data:

```
.device
device code (site dependent, stored in PRINTCAP)
.enddevice
.metadata
metadata text for object 1 for as many lines as it takes
.endmetadata
.pathway
pathway for object 1
.endpath
.metadata
metadata text for object 2 as many lines as it takes
.endmetadata
.pathway
pathway for object 2
.endpath
.
.
.
```

Design Document for the ASCII
GHG Extensions to NELS 1.2

2.1.4 Download.

Activation of the Download option of the Output Request Menu generates a second pulldown menu with the following options:

FTP
KERMIT
X-MODEM
Y-MODEM
Z-MODEM
UUCP

The screens supporting these options are shown in attachment A.

2.1.4.1 FTP.

Selection of the FTP option will generate the following question/answer sequence:

"Target Address"
"Target Userid"
"Target Password"

<O>k, <C>ancel, <H>elp.

The user must fill in the fields and press "O" for OK, "C" for cancel the operation, or "H" for a help explanation. Entering "OK" will cause a check for empty fields and an appropriate error message. Assuming that the fields are filled in, the function will build temporary files for metadata of the objects selected and the associated pathways of the objects. If the "script file" already exists (explained in a moment), then data is appended. After recording all of the data, it will ask the user if transfer should be started now or postponed. If the user answers "Now" then it will then activate FTP and pass the pathway of the metadata and object pointers for transfer to the target address using the target userid and password. At the end of the FTP function, the temporary files are deleted. If the user chooses to postpone transfer, then the file is left open so that additional data may be appended later. The naming convention of the temporary files for metadata is \$ASV3/objectfile_name.metadata. The "script file" will be named \$ASV3/userid.FTP. The format of the "script file" is:

Design Document for the ASCII
GHG Extensions to NELS 1.2

pathway of object 1 metadata
pathway of object 1

.
.
.

pathway of object n metadata
pathway of object n

2.1.4.2 Kermit.

Kermit will use much the same procedures as described for FTP. The difference is that the user will be able to set a variety of variables used by Kermit to regulate the transfer protocols. The variables and the format of the ASCII sequence to get their values will be addressed at a later date.

2.1.4.3 X-Modem, Y-Modem, Z-Modem.

These protocols are closely related. Setting some values controlling Xmodem causes the use of Y-Modem or Z-Modem. The objects and metadata will use much the same procedures as described for FTP.

2.1.4.4 UUCP.

UUCP is a form of Internet Mail. Selection of this option will display a request for the target or "bang" address. UUCP will use much the same procedures for objects and associated metadata files as described for FTP.

2.1.5 Local Copy.

The local copy is provided as a debug tool as much as anything else. It is only available to librarians and not average users. It will create a file named \$HOME/lcopy.datetime with the following data:

```
.metadata
metadata text for object 1 for as many lines as it takes
.endmetadata
.pathway
pathway for object 1
.endpath
.metadata
metadata text for object 2 as many lines as it takes
.endmetadata
.pathway
pathway for object 2
.endpath
.
```

Design Document for the ASCII GHG Extensions to NELS 1.2

A shell script file will then be called to create metadata files and copies of the associated objects in the user's home directory. The same routines which create metadata files used for FTP (and other) transfers will be used following the same file naming conventions. The shell script will not delete the resulting files (unlike the transfer functions).

2.1.6 Exit Processing.

GHG will provide code to be inserted into the exit processing functions of ASCII NELS 1.2 which will query the status of a global variable to determine whether outstanding output requests are pending. If output requests are pending the user will be asked to either start the transmission or cancel the request.

2.2 GHG Shell Scripts.

The majority of the scripts are very simple. The only complication is providing the capability for multiple selections. Multiple selections will be handled in a uniform way. Multiple selections will be accumulated by appending metadata directive (.metadata) followed by the metadata for the object, followed by the pathway directive (.pathway) and the full file pathway to an ascii file in the users home directory. NELS will open this file for append every time the user indicates that a file should be marked for download, copy or print. There will be a separate file for download, copy and print. The GHG scripts will use this file as input (required parameter is filename) and destroy the file upon completion of the function. The media request is a little more complicated because more data is required than just file pathways. The media request will use a set of pseudo directives to separate different types of data needed to detail the files to be transferred and the mailing information.

This approach will provide for cumulative identification of objects as well as allowing the list to be used as a clear text list of instructions if customer service chooses to do the job manually. The File name for this media directive should be userid.datestamp which will allow for storage of multiple requests by the same user if necessary. The ".end" directives are required for any multi-line data values.

Design Document for the ASCII
GHG Extensions to NELS 1.2

The customer service personnel can access these media request files and either use download software from the GHG shell or UNIX utilities to print, TAR, or copy the files to the appropriate devices. The GHG shell will not handle downloads of offline files or files stored on other hosts. Nor will the download handle dumps to tape or print with specific forms. Inclusion of the device code allows for exception handling to detect these types of problems.

3.0 Detailed Interface Description.

All script names are case sensitive.

3.1. Local Copy.

3.1.1 Description.

This routine will copy all files listed within the input file to the user's HOME directory. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be copied because of invalid pathways or lack of permissions.

Script name: Local_Code

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

3.2 Local Print.

3.2.1 Description.

This routine will print all files listed within the input file to the printer requested. The routine will return a code of 0 for successful, 1 for missing parameters, and 2 for partial success. Partial success is where some or all of the files could not be printed because of invalid pathways, lack of permissions, or bad format.

Script name: Local_Print

Required parameters:

character string, full pathway of file list file.

character string, printcap code for required printer.

Return code: 0 for success, greater than 0 is error.

Design Document for the ASCII
GHG Extensions to NELS 1.2

3.3 Copy Requested Objects to Media.

3.3.1 Description.

There are really two classes of scripts here. Most of the PC/MAC compatible media is not a device directly connected to the host. The necessitates Customer Service transferring these files to a PC or MAC first. This may be done by executing a variant of the local_copy script against the files stored in the various "device" directories followed by FTP or Kermit as an interactive user. For those devices directly connected to the host, specific scripts can be executed using the files stored in the appropriate directory. This approach allows customer service to both control the sequence of media copy and the priority. The file naming conventions userid.datetime allows customer service determine how old the outstanding requests are by simply executing a directory command from within UNIX. The routines will return a code of 0 for successful, 1 for missing parameters, 2 bad device code, 3 for partial success.

Script name: Lan_Copy

Function: Creates a copy of requested files in the home directory of the Customer Service representative.

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

Script name: Cart_Tape

Function: Creates a copy of requested files on a 1/4 inch Cartridge Tape.

Required parameters:

character string, full pathway of file list file.

Return code: 0 for success, greater than 0 is error.

The following conventions should be observed for the request_output functions within the GHG Extensions:

Media Request (Copy):

All media requests need to be written to directory "queues". The following directories will contain files named userid.datetime:

```
$ASV3/ibm3.5hd
      /ibm3.5
      /mac3.5
      /ibm5.25hd
      /ibm5.25
      /cart
```

Design Document for the ASCII
GHG Extensions to NELS 1.2

It is expected that Customer Support will periodically scan these directories and process the files appropriately. They can make priority decisions using the date-time stamp convention within the filenames. It is most likely that ibm* and mac* files will FTPed to LAN computers for further processing. They will use the LAN_COPY script to do this.

4.0 Naming Conventions for other "Session" files.

\$HOME/.local_copy	session file for local copy requests.
/.print_req	session file for print requests can support mixed printer types
/.ftp_req	session file for FTP requests (will not support multiple targets)
/.uucp_req	session file for UUCP requests (will not support multiple targets)
/.kermit_req	session file for kermit download requests
/.xmodem_req	session file for xmodem download requests
/.ymodem_req	session file for ymodem download requests
/.zmodem_req	session file for zmodem download requests

5.0 Exit Processing.

The exit processor within the GHG Shell will delete any files of the above types still remaining in the \$HOME directory at logoff with the exception of .local_copy. The startup function within the GHG Shell will delete any .local_copy file found in \$HOME. These files are temporary. This information does not change any specifications within the GHG extensions ICD. It is provided for information only.

Design Document for the ASCII
GHG Extensions to NELS 1.2

Attachment A

Proposed ASCII Screens

April 1, 1992

Attachment A-1

Design Document for the ASCII
GHG Extensions to NELS 1.2

Output Request Screen

c [start] [stop] Copy Selected Objects to computer media.
p [start] [stop] Print Selected Objects to host printers.
d [start] [stop] Download Selected Objects to your site.
l [start] [stop] Local Copy (Copy Objects to \$HOME directory
[Librarians Only]).
h Output Help.
x Cancel

menu command?

Functions:

The user enters selections by a series of numbers. Each object displayed by the Object Browser will have a sequence number. The numbers can be entered as ranges (e.g. 1 - 24) or a series (e.g. 1,3,6,7-12)

Design Document for the ASCII
GHG Extensions to NELS 1.2

Copy Screen (Media Selection)

- 1 3.5 High Density IBM or PC
- 2 3.5 Low Density IBM or PC
- 3 3.5 MAC
- 4 5.25 High Density IBM or PC
- 5 5.25 Low Density IBM or PC
- 6 9 Track 1600 BPI tape (TAR)
- 7 9 Track 1600 BPI tape (ASCII)
- 8 1/4 inch Cartridge Tape
- 9 Hardcopy (14.5 by 11)
- 10 Hardcopy (8.5 by 11)

h Help
x Cancel

Menu Selection?

Function:

The user specifies what type of media to copy the selected objects to.
The order of presentation is totally arbitrary.

Design Document for the ASCII
GHG Extensions to NELS 1.2

Print Screen (Print to Host Printers)

1	print to laser (8.5x11)
2	print to line printer (14.5x8.5)
h	help
x	cancel

Menu Selection?

Function:

The user may select from the available print devices. This is an arbitrary list as an example. The actual devices will be site specific.

document for the ASCII
terminations to NELS 1.2

address).

ion.

available transfer methods. Kermit, X-
ll prompt the user to escape back to
the process. FTP and UUCP will take

Design Document for the ASCII
GHG Extensions to NELS 1.2

Attachment B
Proposed ASCII Help Screens

April 1, 1992

Attachment B-1

Design Document for the ASCII
GHG Extensions to NELS 1.2

Request Output

The Request Output function is provided so that the user may "check out" items from the software repository. The functions provided are:

Copy	Request a copy of an object or objects be provided on a specific media (see Request Output - Copy for more detail)
Download	Download the object and its associated metadata to your system via one of the supported protocols (see Request Output - Download for more detail)
Print	Queue the print of an object and its associated metadata to one of the repository printers. Objects printed out will be forwarded to the requester at the address listed in the Client data file.
Local Copy	Provided as a Librarian Function only. This will make a physical copy of requested objects and their associated metadata in the requesters HOME directory.

The system will record the check out request for each object selected. The repository will notify the requesters (as resources are available) when new versions of object are received or when problem with objects are uncovered. These functions support session requests. This means that after you select the objects and provide whatever data is necessary to complete the output request, you will be asked whether this should be done immediately or later. If you choose later, then object requests will accumulated throughout your session. You may mix print, copy, and download requests and still take advantage of this session capability. The only cost to you is an extended signoff. All stored output request actions are executed prior to logoff.

Main Help Screen for Request Output

Design Document for the ASCII
GHG Extensions to NELS 1.2

Request Output - Now or Later

This message is asking you whether you want to generate your media request or download request now or later. The system will keep track of all the objects you have requested during your session and transmit them at the end of the session at signoff if you would rather do that. Select LATER to postpone the output request execution but to keep track of what was selected. Select NOW and the system will prompt you for appropriate information to execute the output request. The output request is then done in the background while you can continue working.

Help Screen for Now or Later Question

Design Document for the ASCII
GHG Extensions to NELS 1.2

Request Output - Copy

This function is provided so that users may check out objects and receive them on a specified media. The particular media supported at the moment includes:

- 1 3.5 High Density IBM or PC
- 2 3.5 Low Density IBM or PC
- 3 3.5 MAC
- 4 5.25 High Density IBM or PC
- 5 5.25 Low Density IBM or PC
- 6 9 Track 1600 BPI tape (TAR)
- 7 9 Track 1600 BPI tape (ASCII)
- 8 1/4 inch Cartridge Tape
- 9 Hardcopy (14.5 by 11)
- 10 Hardcopy (8.5 by 11)

Multiple objects may be requested at a time. The repository staff will copy the objects on the requested media and ship the results to the requester using mailing information in the Client data file. This is usually the most convenient and fastest way to check objects out of the repository.

***** Note that options 6,7,9,10 are only available when installed at the MountainNet site.

Request Output - Copy - 3.5 High Density IBM or PC

This function requests repository staff to physically copy your selected objects and the associated metadata to a 3 1/2 inch high density (1.44M) diskette readable on an IBM type computer.

Request Output - Copy - 3.5 IBM or PC

This function requests repository staff to physically copy your selected objects and the associated metadata to a 3 1/2 inch low density diskette (720K) readable on an IBM type computer.

Help Screen for Copy (part 1)

Design Document for the ASCII
GHG Extensions to NELS 1.2

Request Output - Copy - 3.5 MAC

This function requests repository staff to physically copy your selected objects and the associated metadata to a 3 1/2 inch diskette readable on an Macintosh type computer.

Request Output - Copy - 5.25 High Density IBM or PC

This function requests repository staff to physically copy your selected objects and the associated metadata to a 5 1/4 inch high density (1.2M) diskette readable on an IBM type computer.

Request Output - Copy - 5.25 IBM or PC

This function requests repository staff to physically copy your selected objects and the associated metadata to a 5 1/4 inch low density diskette (360K) readable on an IBM type computer.

Request Output - Copy - 9 Track 1600 BPI tape (TAR)

This function requests repository staff to physically copy your selected objects and the associated metadata to a 9 track 1600 BPI Reel of computer tape in a TAR format.

Request Output - Copy - 9 Track 1600 BPI tape (ASCII)

This function requests repository staff to physically copy your selected objects and the associated metadata to a 9 track 1600 BPI Reel of computer tape in a ASCII format.

Request Output - Copy - 1/4 inch Cartridge Tape

This function requests repository staff to physically copy your selected objects and the associated metadata to a 1/4 inch Cartridge Tape in a TAR format.

Help Screen for Copy (part 2)

Design Document for the ASCII
GHG Extensions to NELS 1.2

Request Output - Copy - Hardcopy (14.5 by 11)

This function will print objects or at least attempt to print objects on the a HOST printer using a 14.5 x 11 paper size. If the object format is incompatible with the device requested, then the system will return an error. The protocol used to check compatibility is not 100% reliable so the user needs to exercise some care with this function.

Request Output - Copy - Hardcopy (8.5 by 11)

This function will print objects or at least attempt to print objects on the a HOST printer using a 8.5 x 11 paper size (landscape mode). If the object format is incompatible with the device requested, then the system will return an error. The protocol used to check compatibility is not 100% reliable so the user needs to exercise some care with this function.

Help Screen for Copy (part 3)

Design Document for the ASCII
GHG Extensions to NELS 1.2

Request Output - Download

This function is provided to enable users to download selected objects via FTP, Kermit, or Zmodem. In addition objects can be mailed via UUCP. The specific option available within this function are as follows:

- 1 FTP
- 2 KERMIT
- 3 X-MODEM
- 4 Y-MODEM
- 5 Z-MODEM
- 6 UUCP

The system will record the check out request for each object selected. The repository will notify the requesters (as resources are available) when new versions of object are received or when problem with objects are uncovered. See Request Output - Download - option for more specific detail on each option.

Request Output - Download - FTP

This function will prompt for the target site address and your userid/password to access that site. This function transfers files to the target site. You must have a valid userid/password at the site and know the correct internet address of the site.

You will be prompted for your USERID. This is the userid you use to sign the target system.

The PASSWORD is the password that you use to sign on the target system.

The TARGET is the internet address in either alpha (asv3.wvnet.edu) form or numeric (129.71.42.1) form.

After you have entered all the data, press the OK button and the system will attempt to sign on and transfer all the objects you have selected. All objects will be transferred to the home directory of your userid on the target system.

Help Screen for Download (part 1)

Design Document for the ASCII
GHG Extensions to NELS 1.2

Request Output - Download - Kermit, Xmodem, Ymodem, Zmodem

The system will prompt you to escape back to your system and prepare to receive files. The files will be sent as separate pairs (object and metadata) with the metadata files ending with the suffix ".met.". Filenames will be truncated to 8 characters and suffixes to three. Control will return to the Request Output menu upon termination of the transfer. Please note that downloading graphics requires that both the terminal emulation and download protocol on the receiving end be set up to receive 8 bit transfers not 7 bit.

Request Output - Download - UUCP

This function will prompt for the target site address (a so-called bang address). This function transfers files in the background to the target site. This is similar to E-mail. All files will end up in the UUCPPUBLIC directory at the target site. The site administrator can tell you precisely where that is. This directory has general permissions so you can read and retrieve the objects you transmit.

The TARGET is the internet address of the form
user@target!target!target

After you have entered all the data, press the OK button and the system will attempt to sign on and transfer all the objects you have selected.

Help Screen for Download (part 2)

Design Document for the ASCII
GHG Extensions to NELS 1.2

Request Output - Print

This function will print objects or at least attempt to print objects on the requested HOST printer device. If the object format is incompatible with the device requested, then the system will return an error. The protocol used to check compatibility is not 100% reliable so the user needs to exercise some care with this function.

This part is put in to support the demonstration for UHCL

The printer devices currently supported are:

psghg - a Post Script printer located in the GHG Lab
lpghg - a Line Printer located in the GHG facility

Remember that the objects come out on the repository printer so that the repository staff can mail the results to you. They will use whatever mailing information is resident in the Client data file. If you want to check what this file says exit to the shell and access the e-mail function.

Help Screen for Print

Design Document for the ASCII
GHG Extensions to NELS 1.2

Request Output - Local Copy

This function is provided for the convenience of the repository staff. Objects selected will be physically copied to the requester's HOME directory. Non-staff users can not access their HOME directory directly so they will be excluded from using this function.

Help Screen for Local Copy

Design Document for the ASCII
GHG Extensions to NELS 1.2

What would you like to do with the Session File?

This question indicates that you have requested output but delayed its creation. You may select CANCEL which then throws away your output requests or you may select PROCESS which will implement you output requests.

Help Screen for Exit if user still has session files

SOFTWARE DELIVERED

APPENDIX F
DELIVERABLES COVER SHEET

RB.05 GHG Corporation
January 1, 1992

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW #10.

Title of Task: Integrated copy of GHG's extensions into NELS 1.1

Subcontractor: GHG Corporation

Cooperative Agreement No. NCC-9-16

Principal Investigator: E.T. DICKERSON

NASA Technical Monitor: E. FRIDGE

Type of Report: _____

Period Covered by Report _____

Due Date: 04/01/92

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

April 1, 1992

To: M. Rorvig

From: D. Henning

Info: J. McGee
E.T. Dickerson

Subject: Integration of GHG Extensions into NELS 1.1

Russell has completed the modifications of the Browser.C module required to integrate the GHG Extensions into NELS 1.1 as per agreement of March 30, 1992. This code should be given to BTI at the earliest possible opportunity to allow testing of these functions to be included in the current effort. Russell is available to perform the integration of the code or assist in this effort. Please take advantage of this resource while he is available this morning.

NOT Placed under CM
for Revision purposes.

APPENDIX F
DELIVERABLES COVER SHEET

RB.05 GHG Corporation
January 1, 1992

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW # 11

Title of Task: Program management monthly report for the month Jan - Nov '92

Subcontractor: GHG Corporation

Cooperative Agreement No. NCC-9-16

Principal Investigator: E.T. DICKERSON

NASA Technical Monitor: E. FRIDGE

Type of Report: _____

Period Covered by Report 01/92 - 03/92

Due Date: 05/31/92.

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

APPENDIX F
DELIVERABLES COVER SHEET

RB.05 GHG Corporation
January 1, 1992

Research Activity Number RB.05

Subcontract Number: 109

Project/Program: RBSE

Task Deliverable Number of Specific Reference from SOW # 11

Title of Task: Program management monthly report for the month Jan-Mar '92

Subcontractor: GHG Corporation

Cooperative Agreement No. NCC-9-16

Principal Investigator: E.T. DICKERSON

NASA Technical Monitor: E. FRIDGE

Type of Report: _____

Period Covered by Report 01/92-03/92

Due Date: 05/31/92.

Delivered to: RICIS Document Control Department
Box 444
University of Houston-Clear Lake
2700 Bay Area Boulevard
Houston, Texas 77058-1096

RBSE (AdaNET) JANUARY STATUS REPORT

SYSTEM DEFINITION
PRODUCT DEVELOPMENT

GHG Corporation

January 23, 1992

GHG

3.0 SYSTEM DEFINITION

WBS Code	Task Name	Start Date	End Date	1991					1992					1993				
				Mar	May	Jul	Sep	Nov	Jan	Mar	May	Jul	Sep	Nov	Jan	Mar	May	Jul
0	RBSE (AdaNET)	15-Mar-91	31-Jul-95															
3	System Definition	15-Mar-91	31-Jul-95															
3.1	Concept Compliance	15-Mar-91	12-Apr-93															
3.1.1	Concept Analysis	15-Mar-91	10-Feb-93															
3.1.1.1	ASV3 Concept Analysis	15-Mar-91	27-Nov-91															
3.1.2	Define Compliance Criteria	2-Dec-91	12-Mar-93															
3.1.2.1	ASV3 Criteria	2-Dec-91	16-Jan-92															
3.1.3	Report Compliance Criteria	17-Jan-92	12-Apr-93															
3.1.3.1	ASV3 Report	17-Jan-92	19-Mar-92															
3.2	Requirements Definition	1-Apr-91	17-Aug-94															
3.2.1	Functional Req. Definition	1-Apr-91	21-May-92															
3.2.1.1	ASV3 Platform Requirements	1-Apr-91	30-Jul-91															
3.2.1.2	ASV4 Functional Requirements	3-Sep-91	21-May-92															
3.3	Market Analysis - ASV4	9-Jul-91	18-Jul-94															
3.3.1	Plan Effort	9-Jul-91	2-Aug-91															
3.3.2	Research	5-Aug-91	1-Jul-94															
3.3.2.1	ASV4 Research	5-Aug-91	4-Dec-91															
3.3.3	Market Analysis/Report	4-Dec-91	18-Jul-94															
3.3.3.1	Report for ASV4 Requirements	4-Dec-91	10-Jan-92															
3.5	Policies	1-Oct-91	29-Oct-92															
3.5.1	Legal Studies	1-Oct-91	28-May-92															
3.5.1.1	Legal Document Review	1-Oct-91	4-Feb-92															
3.5.3	Repository Standards	2-Dec-91	10-Apr-92															
3.5.3.1	Quality Assurance Standards	2-Dec-91	11-Feb-92															
3.6	Ind. Verification/Validation	13-Jan-92	20-Jul-95															
3.6.1	Test Plans and Procedures	13-Jan-92	30-Jun-95															
3.6.1.1	ASV3 AT Plans/Procedures	13-Jan-92	14-Jul-92															
3.7	Administration - Definition	15-Mar-91	31-Jul-95															
3.7.1	Project Management	15-Mar-91	31-Jul-95															
3.7.2	Reviews/Meetings	15-Mar-91	31-Jul-95															
3.7.2.1	Project Status Meetings	15-Mar-91	31-Jul-95															
3.7.2.2	Technical Reviews	6-Nov-91	7-Nov-91															
3.7.3	Travel	15-Mar-91	31-Jul-95															
3.7.4	Project Support	15-Mar-91	31-Jul-95															
4	Product Development	2-Dec-91	30-Jun-95															
a1	ASV3 Milestones	2-Dec-91	4-Aug-92															
b1	AutoLib 4.0 Delivery	2-Dec-91	2-Dec-91															
6	Quality and Configuration Mgmt	1-Aug-91	31-Jul-95															
6.1	QA and CM Planning	1-Aug-91	18-Nov-91															
6.1.1	Develop CM Plan	1-Aug-91	11-Oct-91															
6.1.2	Develop TQM Plan	1-Oct-91	18-Nov-91															
6.2	Configuration Control	1-Oct-91	31-Jul-95															
6.2.1	System Definition CM	1-Oct-91	31-Jul-95															
6.2.2	Product Development CM	1-Oct-91	31-Jul-95															
6.2.4	Audits	1-Oct-91	31-Jul-95															
6.3	Quality Assurance	18-Nov-91	31-Jul-95															
6.3.1	System Definition QA	18-Nov-91	31-Jul-95															
6.3.2	Product Development QA	18-Nov-91	31-Jul-95															

Scheduled Start Date: 15 March '91 *Percent Complete:* 90
End Date: 27 Nov '91

Status:

- ☐ Document Updated December 11, 1991
- ☐ Suggested Enhancing Scenarios

Hold Up:

- ☐ None if Scenarios Don't Need to be Changed/Added

Scheduled Start Date: 2 Dec '91 *Percent Complete:* 60

End Date: 16 Jan '92

Status:

- ☐ ASV3 Requirements Document Reviewed
- ☐ Hardware Procurement Specifications Reviewed
- ☐ GHG Developed Software

Hold Up:

- ☐ Outstanding Requirements Issues
- ☐ AutoLib SRS Not Reliable Evaluation Source
 - Need a Baselined Copy of AutoLib to Review

Scheduled Start Date: 2 Dec '91 *Percent Complete:* 60

End Date: 16 Jan '92

Status:

- ☐ ASV3 Requirements Document Reviewed
- ☐ Hardware Procurement Specifications Reviewed
- ☐ GHG Developed Software

Hold Up:

- ☐ Outstanding Requirements Issues
- ☐ AutoLib SRS Not Reliable Evaluation Source
 - Need a Baselined Copy of AutoLib to Review

RBSE

System Definition: ASV3 Compliance Report

WBS 3.1.3.1

Scheduled Start Date: 17 Jan '92 *Percent Complete:* 0
End Date: 19 March '92

Status:

☐ Not Started

Hold Up:

☐ Man Power Reallocated

GHG

Scheduled Start Date: 17 Jan '92

Percent Complete: 0

End Date: 19 March '92

Status:

☐ Not Started

Hold Up:

☐ Man Power Reallocated

Scheduled Start Date: 1 April '92 *Percent Complete:* 90
End Date: 30 July '91

Status:

☐ Outstanding Comments From

- MountainNET
- UHCL - University of Houston, Clear Lake

Hold Up:

- ☐ Getting the Forum in Which to Collectively Work These Issues
- ☐ University Needs to Coordinate/Arbitrate

Scheduled Start Date: 3 Sept '91 ***Percent Complete:*** 5
End Date: 21 May '92

Status:

☐ Halted

- GHG Has Made Some Basic Assumptions, But They Appear too Shakey to Proceed

Hold Up:

☐ Concept Document Not Baselined

☐ Direction Not Clear

☐ Man Power Reallocated

- Market Survey Stopped

Scheduled Start Date: 9 July '9 *Percent Complete:* 60

End Date: 4 Dec '91

Status:

- ☐ Market Survey Complete
- ☐ Results Compiled
- ☐ Awaiting Final Review

Hold Up:

- ☐ Man Power Reallocated
- ☐ Subcontractor Terminated

Scheduled Start Date: 1 Oct '91 *Percent Complete:* 0

End Date: 11 Feb '92

Status:

- ☐ Legal Document Review Not Started
- ☐ Quality Assurance Standards Review Not Started

Hold Up:

- ☐ Man Power Reallocated

RBSE

System Definition: ASV3 AT Plans/Procedures

WBS 3.6.1.1

Scheduled Start Date: 13 Jan '92 *Percent Complete:* 0

End Date: 14 July '92

Status:

☐ Not Started

Hold Up:

☐ Man Power Reallocated

GHG

4.0 PRODUCT DEVELOPMENT



WBS Code	Task Name	Start Date	End Date	1991					1992											
				Mar	May	Jul	Sep	Nov	Jan	Mar	May	Jul	Sep	Nov						
0	RBSE (AdaNET)	28-Mar-91	14-Jul-95																	
4	Product Development	28-Mar-91	14-Jul-95																	
4.1	COTS Acquisition	23-Sep-91	7-Sep-93																	
4.1.1	ASV3 Acquisitions	23-Sep-91	28-Feb-92																	
4.1.1.1	Acquisition Plan	23-Sep-91	25-Sep-91																	
4.1.1.2	Solicitation/Evaluation	26-Sep-91	10-Oct-91																	
a1	PDR Prep & Dry Run	16-Oct-91	5-Nov-91																	
b1	Level II PDR	6-Nov-91	6-Nov-91																	
c1	UHCL Procurement	7-Nov-91	21-Feb-92																	
4.2	Rehost to ASV3	23-Sep-91	28-Jul-92																	
a2	4.0 in-house testing	23-Sep-91	18-Oct-91																	
b2	4.0 beta 2 release	21-Oct-91	21-Oct-91																	
c2	4.0 release	22-Nov-91	22-Nov-91																	
d2	Installation at GHG	22-Nov-91	2-Dec-91																	
4.2.1	Detailed Design	30-Oct-91	20-Nov-91																	
4.2.1.2	AutoLib Extension Design	30-Oct-91	20-Nov-91																	
4.2.2	Design Implementation	3-Dec-91	25-Mar-92																	
4.2.2.1	Code	3-Dec-91	11-Feb-92																	
4.7	Transition Support - ASV3/ASV4	21-Nov-91	14-Jul-95																	
4.7.1	Transition Planning	21-Nov-91	19-Jan-95																	
4.7.1.1	ASV3 Transition Plan	21-Nov-91	24-Dec-91																	
4.7.3	Operational Documentation	24-Dec-91	9-Mar-95																	
4.7.3.1	ASV3 Documents	24-Dec-91	6-Apr-92																	
4.8	Administration - Development	28-Mar-91	9-Mar-95																	
4.8.1	Project Management	28-Mar-91	9-Mar-95																	
4.8.2	Reviews/Meetings	28-Mar-91	9-Mar-95																	
4.8.2.1	Project Status Reviews	28-Mar-91	9-Mar-95																	
4.8.2.2	Technical Reviews	28-Mar-91	10-Aug-93																	
4.8.3	Travel	28-Mar-91	9-Mar-95																	
4.8.4	Project Support	28-Mar-91	9-Mar-95																	
										</										

Scheduled Start Date: 23 Sept '91 *Percent Complete:* 30
End Date: 28 Feb '92

Status:

- ☐ UHCL Has This Task/ GHG to Assist
- ☐ Hardware & Software Procurement Specifications Released Dec '92
- ☐ Current Status - Unknown

Hold Up:

- ☐ Unknown

RBSE

Product Development: Rehost AutoLib to ASV3

WBS 4.2 (a2-d2)

Scheduled Start Date: 23 Sept '91 *Percent Complete:* Unknown
End Date: 2 Dec '91

Status:

☐ Unknown

Hold Up:

☐ Re-scope of Effort for AutoLib

CHG

Scheduled Start Date: 30 Oct '91 *Percent Complete:* 40
End Date: 20 Nov '91

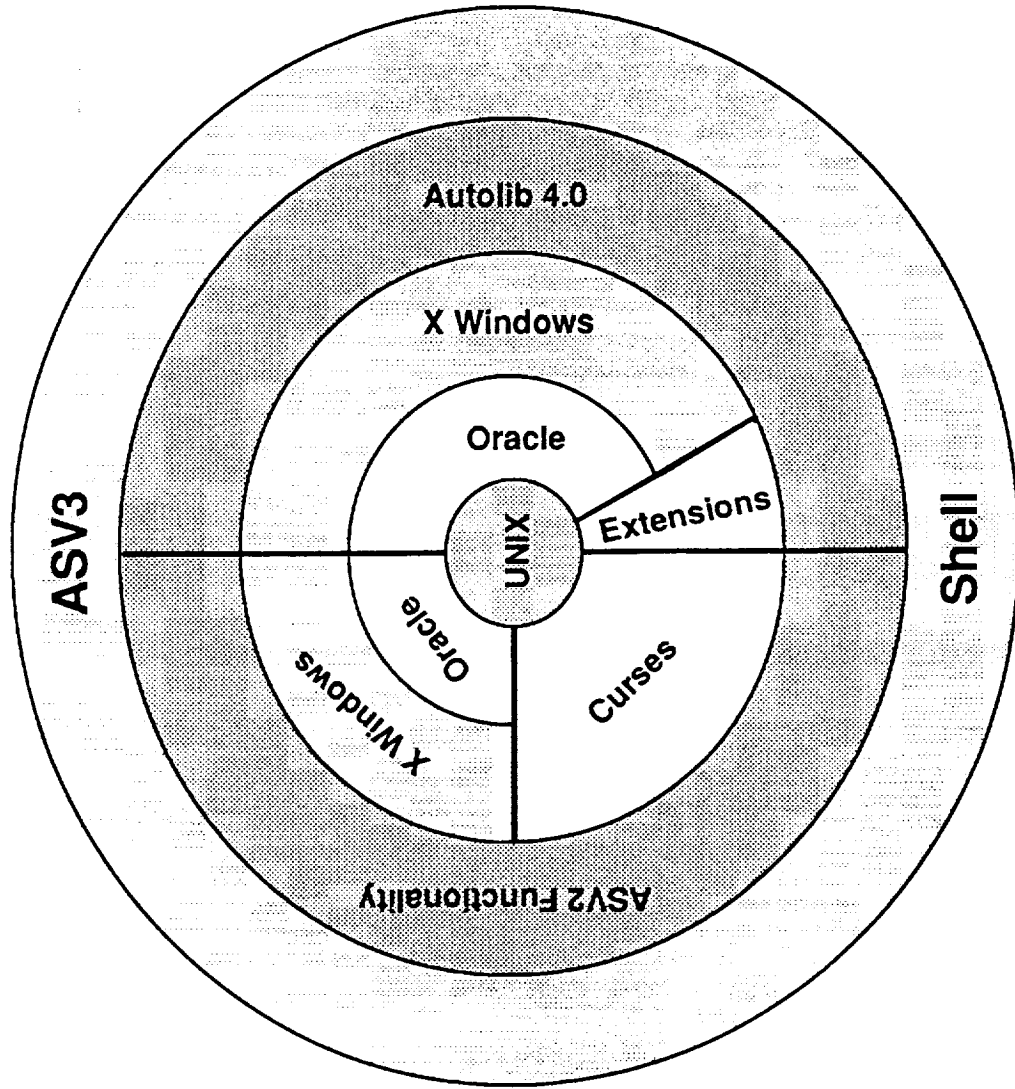
Status:

- ☐ Detailed Requirement Review / Analysis Complete
- ☐ Preliminary Design Complete
- ☐ Prototyping Using Hyperpad

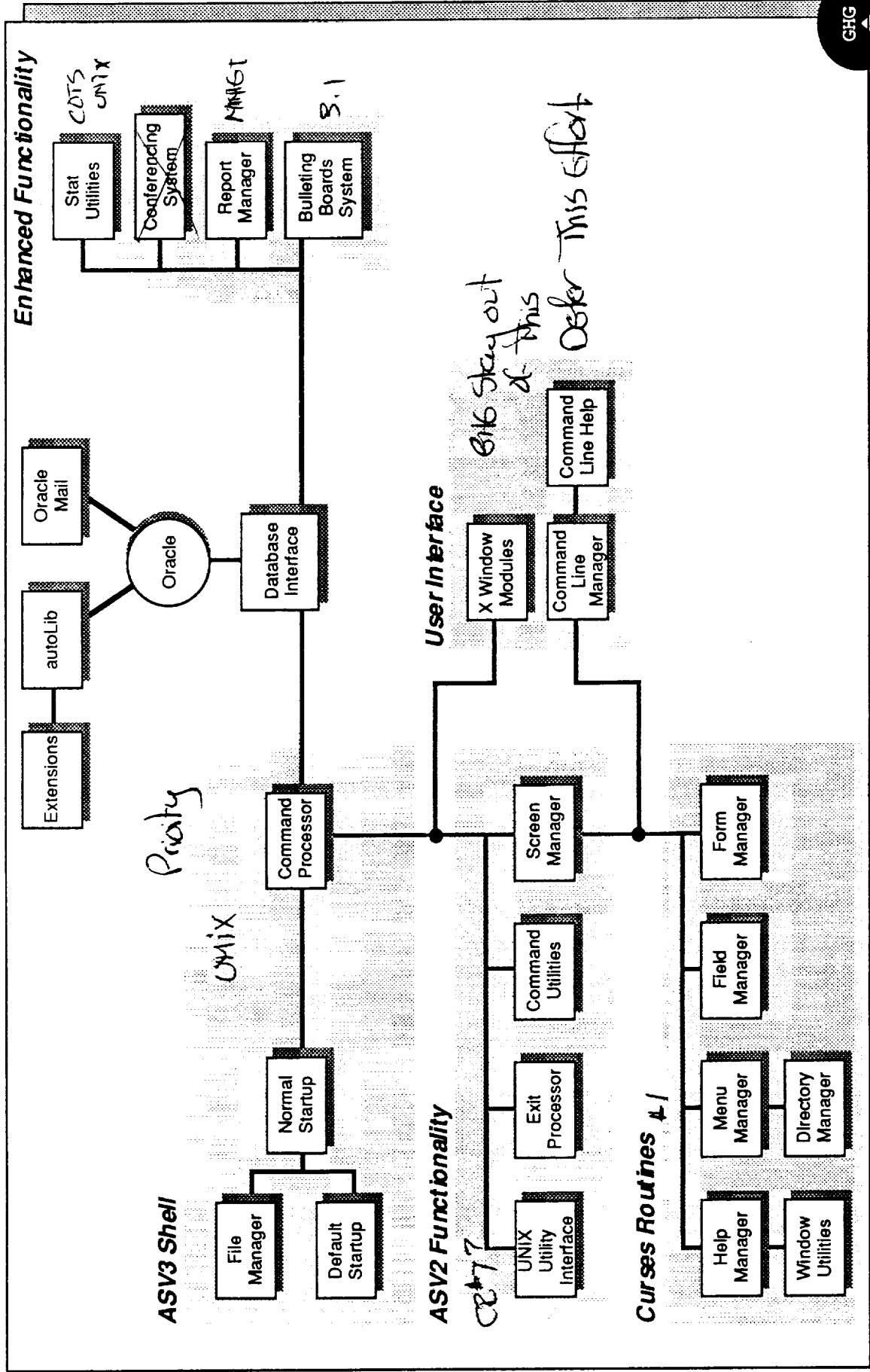
Hold Up:

- ☐ Severe Manpower Impact Due to Funding Cuts
- ☐ Delays In Accessing AutoLib
- ☐ GHG Supporting Other Activities

RBSE ASV3 Repository System



RBSE ASV3 Repository System



RBSE

Product Development: AutoLib Extensions (Code)

WBS 4.2.2.1

Scheduled Start Date: 3 Dec '91 *Percent Complete:* 0
End Date: 11 Feb '92

Status:

☐ Not Started Yet

Hold Up:

☐ Delays in Detail Design Effort Ripple Effect

CHG

Scheduled Start Date: 21 Nov '91 *Percent Complete:* 10

End Date: 24 Dec '91

Status:

- ☐ Planning Effort Began - Approach Presented at Level II PDR
- ☐ Effort Halted

Hold Up:

- ☐ Reduction in Man Power
- ☐ Redirection of Effort

RBSE

Product Development: ASV3 Documents

WBS 4.7.3.1

PRECEDING PAGE BLANK NOT FILMED

Scheduled Start Date: 24 Dec '91 *Percent Complete:* 0

End Date: 6 April '92

Status:

☐ Not Started Yet

Hold Up:

☐ Reduction in Man Power

☐ Cannot Define "End System"

GHG

6.0 QUALITY & CONFIGURATION MANAGEMENT

Scheduled Start Date: 1 Aug '91 ***Percent Complete:*** 75

End Date: 11 Oct '91

Status:

☐ CM Plan Submitted October '91

Hold Up:

☐ Awaiting Comments

Scheduled Start Date: 1 Oct '91 *Percent Complete:* 10

End Date: 18 Nov '91

Status:

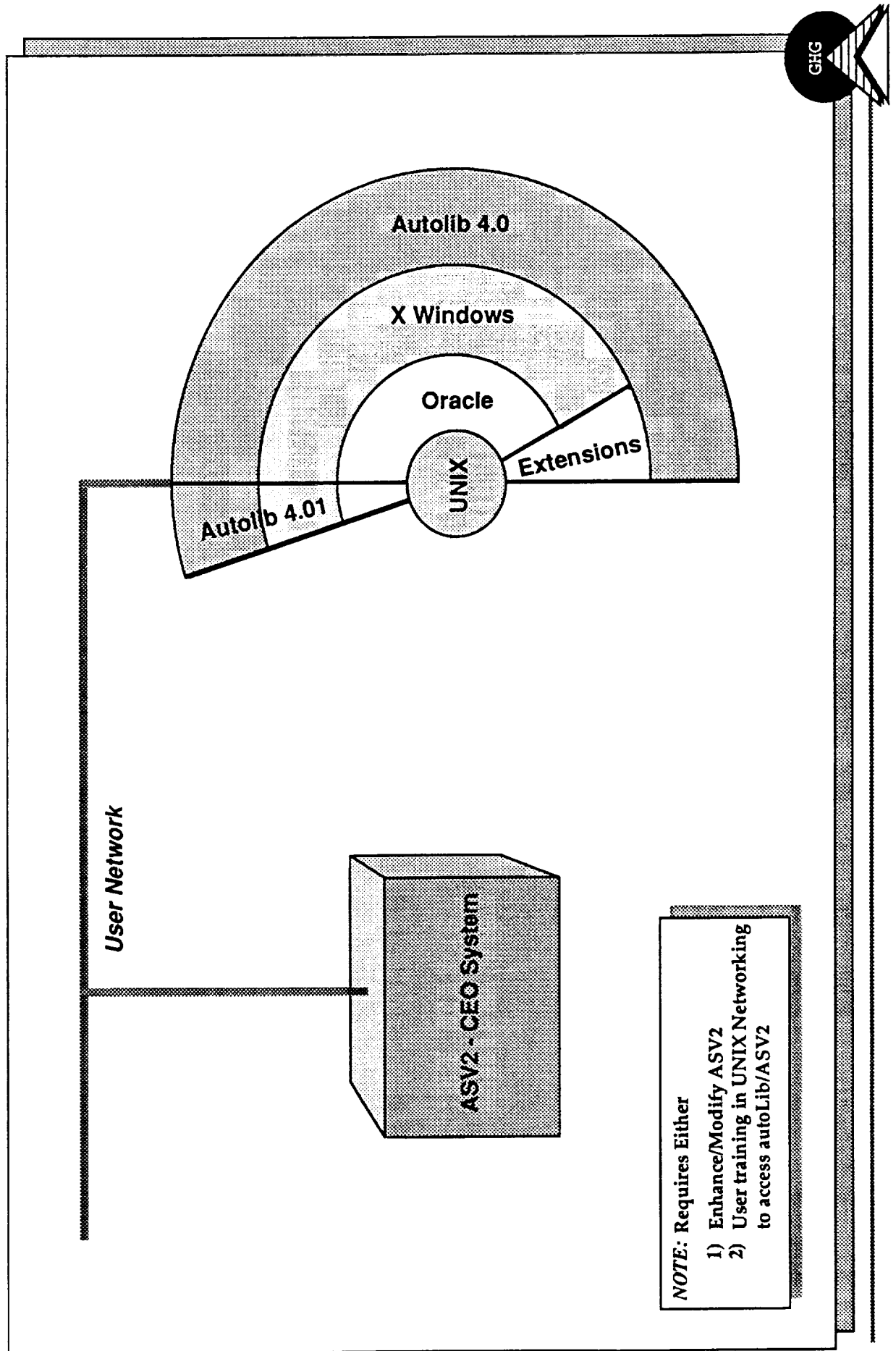
- ☐ Outline Developed
- ☐ Attended TQM Seminars
- ☐ Working with NASA's Total Quality Group
- ☐ Halted

Hold Up:

- ☐ Man Power Reallocated

- ☐ No Contract in Place
- ☐ Recent Re-Direction of Effort
 - Severe Lack of Communication UHCL-GHG
 - Losing Sight of the "Big Picture"
- ☐ GHG's Position/Responsibility
 - What Do You Want Us to Do?
 - Define This Contractually

RBSE ASV2 & autoLib



NOTE: Requires Either

- 1) Enhance/Modify ASV2
- 2) User training in UNIX Networking to access autoLib/ASV2

Hand-out at Tem PMR

Requirement	Reference	ASV3 Subsystem
Users shall be able to return to the most recently specified search criteria and edit them to refine the next search.	ASV3 Requirements Document, paragraph 5.1.1.1.1.f (20Jun91)	autoLib 4.x
ASV3 shall provide the capability to browse the catalog.	ASV3 Requirements Document, paragraph 5.1.1.1.1.g (20Jun91)	autoLib 4.0
Users shall be able to follow links between catalog entries to access entries for related objects.	ASV3 Requirements Document, paragraph 5.1.1.1.1.h (20Jun91)	autoLib 4.x
The users shall be able to determine their current location within the hierarchy.	ASV3 Requirements Document, paragraph 5.1.1.1.1.i (20Jun91)	autoLib 4.0
ASV3 shall provide the capability to view online objects.	ASV3 Requirements Document, paragraph 5.1.1.1.1.j (20Jun91)	autoLib 4.0
ASV3 shall allow the addition of viewing tools for non-ASCII files.	ASV3 Requirements Document, paragraph 5.1.1.1.1.l (20Jun91)	autoLib 4.0
ASV3 shall support the viewing, printing, saving, and retrieving of results from searching the catalog.	ASV3 Requirements Document, paragraph 5.1.1.1.1.m (20Jun91)	autoLib 4.0 partial
Object queuing shall be supported while browsing an object and while examining lists of objects.	ASV3 Requirements Document, paragraph 5.1.1.1.1.p (20Jun91)	autoLib 4.x
ASV3 shall allow the addition, modification, and deletion of objects.	ASV3 Requirements Document, paragraph 5.1.1.1.2.b (20Jun91)	autoLib 4.0 librarian only
Changes to the catalog, whether by addition, modification, or deletion shall be supported in a restricted mode.	ASV3 Requirements Document, paragraph 5.1.1.1.2.c (20Jun91)	autoLib 4.0

Requirement	Reference	ASV3 Subsystem
Updates to catalogs shall be available for review and correction prior to affecting the online system.	ASV3 Requirements Document, paragraph 5.1.1.1.2.c (20Jun91)	autoLib 4.x
ASV3 shall provide the same cataloging capabilities for both online and offline objects.	ASV3 Requirements Document, paragraph 5.1.1.1.2.d (20Jun91)	autoLib 4.0
Library maintenance functions shall not be accessible by the general user.	ASV3 Requirements Document, paragraph 5.1.1.2.h (20Jun91)	autoLib 4.0
The schema for the ASV3 catalog shall accommodate the most useful characteristics of the objects. This includes but is not limited to:	ASV3 Requirements Document, paragraph 5.1.1.3.b (20Jun91)	autoLib 4.0
Author Organization Contact Person Contact Address Contact Telephone Number Electronic Contact Mechanism Electronic Contact Address Title of Product Type (paper, course, conference, source code, design document, etc.) Abstract Keywords Adequate cross references and linkages to locate related components and documents such as: documentation source code object code relevant commercial tools papers proceedings reviews conference announcements sources of training		

Requirement	Reference	ASV3 Subsystem
The schema for the catalog shall accommodate additional characteristics defined, as applicable, for each category.	ASV3 Requirements Document, paragraph 5.1.1.3.c (20Jun91)	autoLib 4.0
The schema for courses and conferences includes but is not limited to:	ASV3 Requirements Document, paragraph 5.1.1.3.1 (20Jun91)	autoLib 4.0
<ul style="list-style-type: none"> Start Date End Date Delete after date location time 		
The schema for software includes but is not limited to:	ASV3 Requirements Document, paragraph 5.1.1.3.1 (20Jun91)	autoLib 4.0
<ul style="list-style-type: none"> Date of Development Date of Last Update Size of online file number of lines of code location language used target environment required hardware & operating system operational restrictions price restrictions on licensing warrantees liabilities 		
The schema for documentation includes but is not limited to:	ASV3 Requirements Document, paragraph 5.1.1.3.1 (20Jun91)	autoLib 4.0
<ul style="list-style-type: none"> Date of Publication date of last update size of online file number of pages location 		
ASV3 shall adopt an open systems architecture based on UNIX and X-windows	ASV3 Requirements Document, paragraph 5.1.2.2.a (20Jun91)	all subsystems

Requirement	Reference	ASV3 Subsystem
ASV3 shall use the Oracle Database to support AutoLib.	ASV3 Requirements Document, paragraph 5.1.2.2.j (20Jun91)	autoLib 4.0
ASV3 shall have a schema which will allow appropriate and consistent cataloging of all existing and anticipated near-term holdings.	ASV3 Requirements Document, paragraph 5.1.2.2.1 (20Jun91)	autoLib 4.0
ASV3 shall support a graphics user interface (GUI) by use of X-windows and the Motif widget set on directly connected terminal with X-windows capability.	ASV3 Requirements Document, paragraph 5.1.2.2.m (20Jun91)	all subsystems
ASV3 shall accommodate a library of 6000 objects expandable to 32000 objects.	ASV3 Requirements Document, paragraph 5.2.1.a (20Jun91)	autoLib 4.0
ASV3 shall be implemented on a UNIX hardware platform supported by the COTS software necessary to host autoLib. AutoLib shall be furnished by NASA. Enhancements to autoLib will be modularized and implemented so as to minimize the cost of maintenance and maximize the ability to accept updates from the baseline version.	ASV3 Requirements Document, paragraph 5.4 (20Jun91)	autoLib 4.0
Connectivity to the ASV3 system shall be automatically terminated after three consecutive, unsuccessful log on attempts.	ASV3 Requirements Document, paragraph 5.1.1.2.e (20Jun91)	UNIX
Inactivity on a user terminal for some period of time (to be determined) shall result in automatic log off of that user.	ASV3 Requirements Document, paragraph 5.1.1.2.f (20Jun91)	UNIX

Requirement	Reference	ASV3 Subsystem
ASV3 shall provide for the administrative creation of new user accounts and definition of user privileges.	ASV3 Requirements Document, paragraph 5.1.1.2.a (20Jun91)	UNIX
Instructions and prompts shall be consistant throughout the system.	ASV3 Requirements Document, paragraph 4.3.d (20Jun91)	All Subsystems dependent on Interface
ASV3 shall provide online help.	ASV3 Requirements Document, paragraph 4.3.c (20Jun91)	All Subsystems
Online assistance shall be available from anywhere within the ASV3 system.	ASV3 Requirements Document, paragraph 4.3.c.2 (20Jun91)	All Subsystems
MESSAGES and prompts shall remain on the screen until acknowledged by the user.	ASV3 Requirements Document, paragraph 4.3.e (20Jun91)	All Subsystems
Displays shall be consistant as to format and to the naming, placement, and assignment of keys.	ASV3 Requirements Document, paragraph 4.3.f (20Jun91)	All Subsystems dependent on Interface
The ANSI terminal interface shall implement all functions using the ANSI X3.154-88 standard. Additional keys may be implemented for convenience, for example, editing, cursor movement, and function keys.	ASV3 Requirements Document, paragraph 4.3.g (20Jun91)	All Subsystems
A consistant set of short cut keys shall be available. For example, users should be able to use a single key to abandon an activity, to obtain help, or to return to the main menu from anywhere in the menu hierarchy.	ASV3 Requirements Document, paragraph 4.3.h (20Jun91)	All Subsystems dependent on Interface
A status line shall indicate which standard and navigational keys are applicable to any location within the menu hierarchy.	ASV3 Requirements Document, paragraph 4.3.i (20Jun91)	Only ANSI Interface

Requirement	Reference	ASV3 Subsystem
Functions of the ANSI terminal emulation will be implemented in such a way as to require the fewest number of keypress/mouse sequences.	ASV3 Requirements Document, paragraph 4.3.j (20Jun91)	All Subsystems dependent on Interface
Data shall be entered directly into those fields requiring user input values.	ASV3 Requirements Document, paragraph 4.3.k (20Jun91)	All Subsystems
All windows, pull-down menus, and messages that overlay parts of a display shall be clearly delineated from the underlying screen contents.	ASV3 Requirements Document, paragraph 4.3.l (20Jun91)	All Subsystems
Operations generally requiring response times greater than 3 seconds shall present a status indication to the user.	ASV3 Requirements Document, paragraph 4.3.m (20Jun91)	All Subsystems
ASV3 shall provide acceptable response when loaded with 5 active users and 2 active librarians. Measurable response time requirements are as follows:	ASV3 Requirements Document, paragraph 5.2.1.b (20Jun91)	All Subsystems
Average standard display time (full screen display time to call up menus, display help, or page a multiple screen display) shall not exceed 1 second for directly connected X-window terminals.		
Average standard display time shall not exceed 8 seconds for an ANSI Terminal connected via 1200 baud modem		
Average keyword search and display of first full screen of results shall not to exceed 5 seconds plus standard display time for the terminal type		

Requirement	Reference	ASV3 Subsystem
The efficiency of file downloading over clean communications lines shall exceed 65% so that a 17,000 byte files can be downloaded via 1200 baud modem within 3 minutes		see previous page
ASV3 shall minimize screen update time by transmission of only those screen portions that have changed.	ASV3 Requirements Document, paragraph 5.2.1.d (20Jun91)	Only Curses option
ASV3 shall use Government furnished and commercial-off-the-shelf (COTS) software as much as possible	ASV3 Requirements Document, paragraph 5.1.2.2.b (20Jun91)	All Subsystems
The AdaNET staff shall accept and process problem reports.	ASV3 Requirements Document, paragraph 5.1.1.4.i (20Jun91)	Problem Report Subsystem
ASV3 shall provide file transfer via Kermit, File Transfer Protocol (FTP), XMODEM, YMODEM, and ZMODEM.	ASV3 Requirements Document, paragraph 4.2.b (20Jun91)	File Transfer Subsystem
ASV3 shall provide easy to use file transfer applications for downloading objects from the library directly to the user's system or PC.	ASV3 Requirements Document, paragraph 5.1.1.1.1.n (20Jun91)	File Transfer Subsystem
ASV3 shall support the Transmission Control Protocol/Internet Protocol (TCP/IP), ISO 8473	ASV3 Requirements Document, paragraph 5.1.2.2.g (20Jun91)	X-Window Interface
ASV3 shall support the Network File System (NFS)	ASV3 Requirements Document, paragraph 5.1.2.2.h (20Jun91)	UNIX
ASV3 shall support File Transfer Protocol (FTP).	ASV3 Requirements Document, paragraph 5.1.2.2.i (20Jun91)	File Transfer Subsystem

Requirement	Reference	ASV3 Subsystem
ASV3 shall provide an electronic mail application supporting communications between all classes of AdaNet users. It shall include the capabilities to create, edit, send, retrieve, acknowledge, delete, reply, and forward messages.	ASV3 Requirements Document, paragraph 5.1.1.1.1.q (20Jun91)	E-mail Subsystem
ASV3 shall provide the capability of defining mailing lists of recipients of electronic mail messages.	ASV3 Requirements Document, paragraph 5.1.1.1.1.r (20Jun91)	E-Mail Subsystem
ASV3 shall support the distribution of electronic mail using complete network addresses. In addition, the users shall have the capability to define, update, and delete aliases for use in simplifying the addressing of electronic mail.	ASV3 Requirements Document, paragraph 5.1.1.1.1.s (20Jun91)	E-Mail Subsystem
ASV3 shall use a COTS electronic mail utility	ASV3 Requirements Document, paragraph 5.1.2.2.n (20Jun91)	E-Mail Subsystem
ASV3 shall have spare capacity to accommodate 10 users and 4 librarians with no more than 50% degradation in response time	ASV3 Requirements Document, paragraph 5.2.1.c (20Jun91)	All Subsystems
File Preparation Tools shall include but not be limited to, the entry, editing, and spell checking of object text files.	ASV3 Requirements Document, paragraph 5.1.1.1.2.e (20Jun91)	Utilities Subsystem
ASV3 shall use COTS software to support text file preparation, maintenance, and spell checking	ASV3 Requirements Document, paragraph 5.1.2.2.e (20Jun91)	Utilities Subsystem

Requirement	Reference	ASV3 Subsystem
.ASV3 shall provide word processing and publishing capabilities to directly connected UNIX workstations, ASV3 host connected IBM PCs and ASV3 host connected Macintoshes	ASV3 Requirements Document, paragraph 5.1.2.2.f (20Jun91)	UNIX
ASV3 shall provide reports of the following library statistics:	ASV3 Requirements Document, paragraph 5.1.1.1.2.f (20Jun91)	Reports Subsystem Stat Utilities
<ul style="list-style-type: none"> File sizes or objects with summarys by collection User statistics and demographics including: <ul style="list-style-type: none"> account activation date sessions per user files accessed per session date of the first session date of the last session geographic location organizational affiliation Object usage including: <ul style="list-style-type: none"> number of accesses for each object number of accesses for each collection <p>This information shall persist across modifications to the object and/or its catalog information.</p> <p>The number of additions, archivals, and deletions from the catalog and classification of holding such as reusable software or conference announcements</p>		
ASV3 shall provide log on and password capabilities as a mechanism for ensuring user privacy and the safeguarding of system files.	ASV3 Requirements Document, paragraph 5.1.1.2.b (20Jun91)	Utilities Subsystem

Requirement	Reference	ASV3 Subsystem
ASV3 shall allow users to change their passwords.	ASV3 Requirements Document, paragraph 5.1.1.2.c (20Jun91)	Utilities Subsystem
No ASV3 functions shall be available prior to successful log on.	ASV3 Requirements Document, paragraph 5.1.1.2.d (20Jun91)	UNIX
ASV3 shall provide file transfer applications for uploading files to the ASV3 host machine.	ASV3 Requirements Document, paragraph 5.1.1.1.2.a (20Jun91)	Submit Subsystem
File Preparation Tools shall exist to support the catalog definition of information and object contents.	ASV3 Requirements Document, paragraph 5.1.1.1.2.e (20Jun91)	Submit Subsystem
ASV3 shall support searching within online documents for the occurrence of a string of characters.	ASV3 Requirements Document, paragraph 5.1.1.1.1.k (20Jun91)	Text Search Subsystem
A standard, main menu shall always be presented to the user after log on.	ASV3 Requirements Document, paragraph 4.3.b (20Jun91)	Startup routines
Online help shall be available as a command line option.	ASV3 Requirements Document, paragraph 4.3.c.1 (20Jun91)	All Subsystems
ASV3 shall display a disclaimer message and receive user acknowledgement prior to presenting any options for service to the user., MountainNet UNIX	ASV3 Requirements Document, paragraph 5.1.1.2.g (20Jun91)	Startup routines
A mechanism shall exist to allow users to queue the printing of one or more documents and the copying of one or more objects to external media for physical distribution.	ASV3 Requirements Document, paragraph 5.1.1.1.1.o (20Jun91)	Media Request Subsystem

Requirement	Reference	ASV3 Subsystem
ASV3 shall provide the capability to collect data, measure performance, and support performance tuning of the system.	ASV3 Requirements Document, paragraph 5.1.1.1.2.g (20Jun91)	Stat Utilities
The ASV3 system shall support the addition of an Ada Compiler for eventual maintenance of software objects written in the Ada language	ASV3 Requirements Document, paragraph 5.1.2.2.c (20Jun91)	UNIX
ASV3 shall have a C compiler to support maintenance of autoLib and the development and maintenance of its tools.	ASV3 Requirements Document, paragraph 5.1.2.2.d (20Jun91)	UNIX

RBSE (AdaNET) FEBRUARY STATUS REPORT

SYSTEM DEFINITION
PRODUCT DEVELOPMENT

GHG Corporation

February 13, 1992

GHG

- ☐ Prioritization of PMP Tasks Given 23 Jan '92
- ☐ SOW Issued 30 Jan '92
 - Specific Delivery Dates for Software Components
- ☐ Meeting at JSC on 4 Feb '92
 - Direction Given
 - Bulk Loader Prioritized
- ☐ GHG Will Provide Response to SOW by 21 Feb '92



GHG/MountainNET - Consensus with three Outstanding Issues


- The Use of the Review of ASV2 Software and Documentation as an Applicable Document
- Potential Liabilities not Specifying Detailed LMS Requirements
- Chapter 11 Forecast Section to be Updated and Rewritten

RBSE *ASV3 Requirements Document*

GHG/UHCL - Dr. McKay's Comments Addressed in 2/12 Fax

- GOSIP Compliance Issues Inconsistent with Procurement Specification
- More General Requirements or Establish Goals
- Using ASV4 Concept Document as a Source of ASV3 Requirements


 **Print, Copy and Download ICD Delivered 2/11/92**

 **Curses Routines**

- Cannot Be Separated

 **ASV2 Functionality**

- Screen Manager

 **AutoLib Extensions**

- Routines Written Not Tested

**Bulk Loader**

- Preliminary Investigation
- Requires Purchase of DG Toolkit and Documentation
- Steep Learning Curve
- Not a Trivial Effort



RBSE (AdaNET) MARCH STATUS REPORT

GHG Corporation

March 12, 1992



- ❏ ICD Developed/Delivered to Barrios Feb. 4, 1992
- ❏ Received Questions on Feb. 21
- ❏ Meeting on 28th to Resolve ICD
- ❏ GHG will do Work and Provide Code to NASA for Integration
- ❏ Adds Scope and Impacts Schedule



**Includes:**

- Curses Routines
- Command Line Manager
- Command Line Help
- Command Processor

**Cannot Start without AutoLib ASCII Code**



Barrios will Supply Data For History Tables



Reports can be Developed Using Oracle Report Writer

- ❑ Cannot Load Objects Into AutoLib Yet For Testing
- ❑ ASCII Version Not Even Started Yet
 - GHG Cannot Wait Until June
- ❑ Porting to New Platform
- ❑ Hardware Procurement Delivery



- ☐ Memorandums of Agreement Being Developed
 - Will Fax on Monday 3/16/92
- ☐ Supplimenting AdaNET Staff for Help in Developing Documentation
- ☐ Requirements Document Baselined 3/11/92

[illegible]